



En este módulo

1. Cómo "recuerda" una IA
2. La ventana de contexto
3. Tipos de memoria en IA
4. CLAUDE.md: instrucciones persistentes
5. MEMORY.md: memoria automática
6. Custom Instructions en ChatGPT
7. Técnicas para sesiones largas
8. Límites y soluciones

Cómo "recuerda" una IA

La IA no tiene memoria como un humano. Cada conversación es independiente por defecto. Cuando cierras el chat y abres otro, la IA empieza de cero. No recuerda nada de conversaciones anteriores a menos que uses mecanismos específicos.

Dentro de una conversación, la IA "recuerda" todo lo que se ha dicho. Pero esa memoria tiene un límite: la ventana de contexto.

La ventana de contexto

La ventana de contexto es la cantidad máxima de texto que la IA puede procesar en una sola conversación. Se mide en tokens (aproximadamente 0,75 palabras por token en español).

Ventanas de contexto (2026):

ChatGPT: varía según plan y modelo
 Claude Sonnet 5.5: 1M tokens (~555.000 palabras, API)
 Claude Opus 5.5: 1M tokens (~555.000 palabras, API)
 Claude Haiku 4.5: 200K tokens (~150.000 palabras)
 Gemini (AI Pro): 1M tokens

Para referencia:

- Un email: ~200 palabras (~270 tokens)
 - Un artículo de blog: ~2.000 palabras (~2.700 tokens)
 - Un libro: ~80.000 palabras (~107.000 tokens)

Importante

Que la ventana sea grande no significa que debas llenarla. La calidad de las respuestas baja cuando el contexto es muy largo. Usa solo la información relevante para la tarea actual.

Tipos de memoria en IA

- **Memoria de conversación:** Lo que se ha dicho en el chat actual. Desaparece al cerrar.
- **Instrucciones personalizadas:** Custom Instructions (ChatGPT) o Project Instructions (Claude). Persisten entre conversaciones.
- **Memoria automática:** ChatGPT Memory o MEMORY.md de Claude Code. La IA guarda hechos clave automáticamente.
- **Documentos de referencia:** Archivos subidos a Projects o Knowledge de GPTs. Siempre disponibles.
- **Archivos de configuración:** CLAUDE.md en Claude Code. Se carga automáticamente en cada sesión.

CLAUDE.md: instrucciones persistentes

CLAUDE.md es un archivo Markdown que Claude Code lee automáticamente al iniciar cualquier sesión en ese directorio. Es el equivalente a un manual de empleado para la IA.

```

Ubicación:
~/claude/CLAUDE.md      → Global (todas las sesiones)
./CLAUDE.md             → Por proyecto (en la raíz del repo)

Contenido recomendado:
# CLAUDE.md

## Quién soy
Desarrollador full-stack, startup SaaS B2B.

## Stack tecnológico
- Backend: FastAPI + Python 3.11
- Frontend: Next.js 14 + TypeScript
- BD: Supabase (PostgreSQL)
- Deploy: Cloudflare Pages + Hetzner

## Convenciones de código
- Nombres de variables en snake_case (Python)
- Nombres de componentes en PascalCase (React)
- Tests obligatorios para funciones críticas
- Commits en español, formato convencional

## Reglas
- NUNCA modificar archivos .env
- SIEMPRE pedir confirmación antes de borrar archivos
- Responder en español por defecto

```

MEMORY.md: memoria automática

Claude Code mantiene un archivo MEMORY.md que se actualiza automáticamente con información importante de cada sesión.

```
~/claude/projects/<proyecto>/memory/MEMORY.md
```

```

Este archivo contiene:
- Decisiones técnicas tomadas
- Bugs encontrados y resueltos
- Preferencias del usuario descubiertas
- Referencias a archivos y rutas importantes
- Estado de tareas en curso

```

Puedes editar MEMORY.md manualmente para añadir o eliminar información. También puedes pedirle a Claude Code que "recuerde" algo específico.

Custom Instructions en ChatGPT

ChatGPT permite personalizar sus respuestas en Ajustes > Personalización. Conviene cubrir dos bloques:

- **"Qué quieres que ChatGPT sepa de ti":** Tu contexto personal y profesional.
- **"Cómo quieres que responda":** Estilo, formato, restricciones.

Ejemplo "Qué sepa de mí":

Soy product manager en una fintech española.

Trabajo con equipos de 5-10 personas.

Uso Notion, Slack, Figma y GitHub.

Mi audiencia principal son founders y CTOs.

Ejemplo "Cómo responder":

- Responde en español
- Sé directo, sin rodeos
- Usa bullets para listas, no párrafos largos
- Si no sabes algo, dílo claramente
- Máximo 300 palabras por respuesta
- Incluye ejemplos concretos siempre que sea posible

Técnicas para sesiones largas

Cuando una conversación se alarga, la IA puede perder foco. Estas técnicas ayudan:

- **Resúmenes intermedios:** Cada 10-15 mensajes, pide un resumen de lo decidido.
- **Anclas de contexto:** "Recuerda que estamos trabajando en X y hemos decidido Y".
- **Sesión nueva con resumen:** Si la conversación es muy larga, empieza una nueva pegando el resumen.
- **Archivos de estado:** Guarda decisiones en un archivo que puedas subir a la siguiente sesión.

Límites y soluciones

Los principales límites de la memoria en IA y cómo mitigarlos:

Límite: No recuerda entre sesiones

Solución: CLAUDE.md + MEMORY.md + Projects

Límite: Pierde foco en conversaciones largas

Solución: Resúmenes intermedios + sesiones cortas

Límite: No puede acceder a tus datos

Solución: MCP + documentos de referencia

Límite: Inventa datos que no tiene

Solución: "Si no lo sabes, di 'no tengo esa información'"

Regla práctica

Trata la configuración de memoria como una inversión. Dedicar 30 minutos a configurar CLAUDE.md y Custom Instructions. Esos 30 minutos te ahorrarán horas cada semana.

Ejemplo completo de CLAUDE.md profesional

```
# CLAUDE.md — Mi Proyecto SaaS

## Quién soy
Full-stack developer, startup B2B de gestión de inventario.
2 años de experiencia con Python y React.

## Stack tecnológico
- Backend: FastAPI + Python 3.12
- Frontend: Next.js 14 + TypeScript + Tailwind
- BD: Supabase (PostgreSQL + RLS)
- Deploy: Cloudflare Pages (frontend) + Hetzner (backend)
- CI/CD: GitHub Actions

## Convenciones de código
- Python: snake_case, type hints obligatorios, docstrings
- TypeScript: camelCase, interfaces sobre types
- Commits: conventional commits en español
- Tests: pytest (backend), vitest (frontend)
- Lint: ruff (Python), eslint + prettier (TS)

## Reglas estrictas
- NUNCA modificar archivos .env ni secrets
- NUNCA hacer git push --force a main
- SIEMPRE pedir confirmación antes de borrar archivos
- SIEMPRE usar RLS en nuevas tablas Supabase
- Responder en español por defecto

## Base de datos
- Multi-tenant con RLS (tenant_id en todas las tablas)
- Migraciones con Alembic
- Naming: tablas en plural, columnas en snake_case

## Estructura del proyecto
- /api          → FastAPI endpoints
- /api/models   → SQLAlchemy models
- /api/services → Business logic
- /web          → Next.js frontend
- /web/components → React components
```

Tabla: tipos de memoria y cuándo usar cada uno

TIPO	PERSISTE ENTRE SESIONES	CONFIGURACIÓN	MEJOR PARA
Conversación	No	Automática	Tareas puntuales, brainstorming
CLAUDE.md (proyecto)	Sí	Archivo en raíz del repo	Reglas del proyecto, stack, convenciones
CLAUDE.md (global)	Sí	~/.claude/CLAUDE.md	Preferencias personales, estilo
MEMORY.md	Sí	Automática + manual	Decisiones, bugs, referencias
Projects (Claude)	Sí	Interfaz web	Documentos de referencia, instrucciones
Custom Instructions	Sí	Configuración ChatGPT	Contexto personal, reglas de estilo
GPT Knowledge	Sí	Subir archivos al GPT	Base de conocimiento estática

Técnicas avanzadas de gestión de contexto

1. RESUMEN PROGRESIVO

Cada 15-20 mensajes en una conversación larga:

"Resume todo lo que hemos decidido hasta ahora en bullets.

Incluye: decisiones tomadas, tareas completadas, pendientes."

Guarda ese resumen. Si la conversación se degrada, inicia una nueva pegando el resumen como contexto inicial.

2. ARCHIVOS DE ESTADO

Para proyectos complejos, mantiene un archivo estado.md:

- Decisiones tomadas (con fecha)
- Bugs encontrados y resueltos
- Tareas pendientes
- Referencias a archivos clave

Súbelo al Project o pégalo al inicio de cada sesión.

3. ETIQUETADO DE PRIORIDAD

En instrucciones, usa jerarquía clara:

"CRÍTICO (rompe cosas si no se cumple):

- No borrar archivos sin confirmar
- No pushear a main directamente

IMPORTANTE (afecta calidad):

- Type hints en Python
- Tests para funciones nuevas

PREFERENCIA (nice to have):

- Docstrings en todas las funciones
- Comentarios en español"

Errores comunes con memoria y contexto

- **CLAUDE.md demasiado largo:** Si supera las 2.000 palabras, la IA pierde foco en las reglas menos importantes. Prioriza: las 5 reglas más críticas primero, el resto como referencia.
- **No actualizar después de cambios:** Si migraste de MySQL a PostgreSQL y el CLAUDE.md sigue diciendo MySQL, la IA generará código para el stack equivocado.
- **Pegar documentos enteros como contexto:** En vez de pegar 50 páginas, resume los puntos relevantes o sube el documento a Projects/Knowledge. El contexto debe ser relevante, no masivo.
- **Olvidar la ventana de contexto:** Aunque sea grande (hasta 1M tokens), la calidad baja con contextos muy largos. Usa solo la información necesaria para la tarea actual.

Consejo: Configura tu CLAUDE.md el primer día que uses Claude Code en un proyecto.

Dedícale 20 minutos a escribir stack, convenciones y reglas críticas. Es la inversión con mayor

retorno: cada sesión futura arrancará con contexto correcto en vez de explicaciones repetidas.

IAcademy · iacademy.com · © 2026 Riskitera S.L.U.

MT05: Memoria y contexto · Avanzado

¿Dudas? Usa el tutor IA en iacademy.com/course/mt05/