O que é o REA e o que você vai conseguir

Sabe aquela funcionalidade de um aplicativo que você queria ter no seu projeto? O **REA (Reverse Engineer Anything)** é um projeto aberto, com licença MIT, que dá ao seu agente de IA (Claude Code, Codex, Cursor, Gemini CLI, Grok Build e outros) ferramentas para **investigar como um aplicativo funciona por dentro** — mesmo sem o código-fonte — e depois **construir uma versão própria** no seu projeto.

O ciclo é sempre o mesmo: **você aponta a funcionalidade → o agente investiga com o REA → mostra as evidências → escreve e testa a sua versão**. A análise roda no seu computador e cada conclusão vem com a prova e os limites encontrados.

- **Tempo:** cerca de 15 minutos até o primeiro resultado.
- **Nível:** iniciante que já abriu um terminal uma vez.
- **Custo:** o REA é gratuito; o uso do seu agente (Claude, Codex etc.) segue o plano que você já tem.

> Exemplo real do próprio projeto: o REA rastreou como o aplicativo do Notion leva o conteúdo copiado até a área de transferência, passando pelas camadas internas do app. É esse tipo de pergunta que ele responde.

1. Prepare o computador

1. **Node.js e npm.** O REA pede Node.js **22.x (22.23.4 ou mais novo), 24.x (24.22.1 ou mais novo) ou 26.12.1+**. Confira com o comando abaixo no terminal. Se aparecer erro ou uma versão menor, instale a versão LTS em nodejs.org e abra um terminal novo.
2. **Um agente de código instalado e logado**: Claude Code ou Codex (este guia mostra os dois). O chat comum do navegador não serve — o REA conversa com o agente que roda no seu computador.
3. **Uma pasta de teste** (ex.: `meu-projeto`) ou uma cópia do seu projeto. Assim nada do que é importante muda sem você revisar.

Para começar, você **não precisa** de Hopper, Ghidra ou IDA: sites e aplicativos feitos em JavaScript/Electron (como Notion, Obsidian, Slack e VS Code no desktop) são analisados sem eles. No Linux, alguns fluxos exigem um `ps` compatível com procps.

Terminal

```
node -v
```

2. Instale o REA no Claude Code e/ou no Codex

1. Abra o **terminal** (no Mac: Terminal ou iTerm; no Windows: PowerShell) e rode o instalador oficial abaixo.
2. Na lista de agentes, marque **Claude Code**, **Codex** ou os dois (barra de espaço marca, Enter confirma).
3. O instalador mostra o que vai alterar. Leia e aprove. Ele faz backup das suas configurações antes de mexer.
4. Se ele oferecer instalar o Hopper, pode recusar por enquanto — só é necessário para binários nativos.
5. **Feche e abra de novo** o Claude Code / Codex. Sem reiniciar, o agente não enxerga o REA.

Usa Claude Desktop também? Feche-o completamente antes de aprovar as mudanças. Tem configurações personalizadas no Codex? Veja a seção de configuração de clientes antes.

Terminal

```
npx rea-agents setup
```

3. Confirme que o agente enxerga o REA

Abra o Claude Code ou o Codex **dentro da sua pasta de teste** e cole o pedido abaixo. Você quer ver a lista real de ferramentas — não só um “sim, está instalado”.

Se o agente disser que não encontrou o REA: confira se marcou o agente certo no instalador, reinicie a sessão e veja a seção **Problemas comuns**, mais abaixo.

Pedido para copiar

```
Confira se o REA está disponível nesta sessão. Liste as ferramentas do REA que
você realmente consegue acessar e diga, em uma frase cada, para que servem. Não
analise nada ainda. Se não encontrar o REA, mostre o erro exato.
```

4. Seu primeiro resultado em 10 minutos

Comece com um aplicativo instalado no seu computador e **uma funcionalidade pequena**. Boas primeiras escolhas: a **busca** do Obsidian, a lista de **atalhos** do VS Code, o **copiar e colar** do Notion, o **modo escuro** de um app.

Cole o pedido trocando os colchetes. O agente vai localizar o app, investigar com o REA, mostrar as evidências (arquivos, funções, fluxo) e então propor a sua versão.

O que esperar: uma explicação de como a funcionalidade funciona, a lista de evidências, o que ficou desconhecido e um plano (ou código) para o seu projeto.

Pedido para copiar

```
Entenda como funciona [A BUSCA] no aplicativo [NOME DO APP] instalado neste
computador. Use o REA para investigar, me mostre as evidências do que encontrou e
o que ficou desconhecido. Depois proponha como construir uma funcionalidade
parecida no meu projeto, com código e textos próprios.
```

5. O que dá para analisar

- **Apps desktop em JavaScript/Electron** (Notion, Obsidian, Slack, VS Code…): módulos, rotas, comunicação interna. Não precisa de ferramenta extra.
- **Sites:** estrutura da página, scripts e requisições observadas. Precisa de um navegador da família Chrome.
- **Apps Android (APK):** manifesto, classes e métodos. Precisa do JADX e de um JDK.
- **Programas .NET:** metadados e instruções, sem ferramenta extra.
- **Binários nativos e jogos:** pseudocódigo e chamadas. Precisa de Hopper, Ghidra ou IDA.
- **Outros:** arquivos de modelos de IA (GGUF/safetensors), pacotes Python empacotados, bancos SQLite, firmware e mais.

Dica: comece por app Electron ou site. É onde o resultado aparece mais rápido. A lista completa e os requisitos de cada tipo estão no README oficial.

6. Aprofunde: investigue uma funcionalidade de cada vez

Para funcionalidades maiores, separe **investigar** de **construir**. Primeiro peça só o mapa: tela, dados, regras e estados. Quanto menor o recorte, melhor a resposta.

Lembre: o que roda no servidor do aplicativo (banco de dados, regras internas na nuvem) não está no seu computador. O REA mostra o que existe no app e no tráfego que você observa — e diz o que ficou de fora.

Pedido para copiar

```
Quero entender [FUNCIONALIDADE] do [APP/SITE] para criar uma versão própria no meu projeto. Use o REA para investigar e me entregue uma tabela com quatro colunas: o que observou, a evidência (arquivo, função ou requisição), sua hipótese e o que ficou desconhecido. Separe o que roda no app do que provavelmente roda no servidor. Ainda não escreva código.
```

7. Construa a sua versão no seu projeto

Com o mapa em mãos, peça a implementação **dentro do seu projeto**, seguindo o padrão que ele já usa. O agente deve mostrar quais arquivos vai alterar antes de editar.

Exemplo guiado: depois de investigar a busca do Obsidian, peça uma busca com filtro por título e conteúdo no seu app de anotações, com destaque do termo encontrado e atalho de teclado.

Pedido para copiar

```
Com base no mapa que você fez, implemente uma versão própria de [FUNCIONALIDADE]
no meu projeto. Leia a estrutura atual e siga o padrão existente. Antes de
editar, liste os arquivos que vai alterar. Use código, textos e visual próprios —
nada copiado do app original. Ao final, explique como rodar, como testar e como
desfazer as alterações.
```

8. Teste antes de usar de verdade

Peça que o agente prove que funciona — não apenas que diga que funciona. Uma boa lista de conferência:

1. O caso principal funciona com um exemplo real seu.
2. Um caso que não deveria funcionar é recusado (entrada vazia, termo inexistente).
3. Nada quebrou no resto do projeto.
4. Funciona no celular, se for uma tela.
5. Você sabe como desfazer a mudança.

Pedido para copiar

```
Teste a implementação: rode os testes que conseguir, mostre o resultado de cada
um e o que não foi possível testar. Inclua um caso que deve funcionar, um que
deve falhar e a confirmação de que o restante do projeto continua funcionando.
Não marque como aprovado o que você não executou.
```

9. Bônus: use o REA direto pelo terminal

Quer só olhar por dentro de um app JavaScript/Electron, sem agente? Rode o comando abaixo trocando o caminho pela pasta do app extraído (ou pelo arquivo .asar). O resultado lista módulos, imports e as fronteiras do Electron, com evidências.

Para instalar o comando `rea` de forma permanente: `npm install --global rea-agents`. Depois, `rea --help` mostra tudo e **`rea update`** atualiza — o projeto lança correções com frequência, vale atualizar sempre.

Terminal

```
npx -y rea-agents@latest analyze-javascript-application /caminho/absoluto/do/app
--json
```

Problemas comuns

- **“O agente não encontra o REA”:** reinicie o Claude Code/Codex; confira se marcou esse agente no instalador; rode `npx rea-agents doctor` para ver o diagnóstico.
- **Erro de versão do Node:** instale o Node LTS em nodejs.org, feche e abra o terminal e rode `node -v` de novo.
- **Claude Desktop não mostra o REA:** feche o app completamente (não só a janela) e abra de novo.
- **Pede Hopper/Ghidra:** só para binários nativos. Para apps Electron e sites, não é necessário.
- **Quer remover:** veja a seção diagnóstico, atualização e remoção da documentação oficial.

Use com responsabilidade

- Analise apenas apps e contas que você tem direito de usar, respeitando termos de uso, direitos autorais e privacidade.
- Construa a **sua** versão: código, textos, marca e visual próprios. Não copie código proprietário nem identidade de outra empresa.
- Não use para burlar login, pagamento ou bloqueios.
- A licença MIT vale para o REA, não para o app que você analisa.
- A análise roda no seu computador, mas o agente envia trechos ao modelo de IA conforme a configuração dele. Não use dados privados de clientes.

Uso comercial com dúvida sobre o alvo? Leia os termos do aplicativo antes. Este material é educativo e não substitui orientação jurídica.

Fontes oficiais e próximos passos

Comandos e requisitos conferidos no README e na documentação oficial em **24/10/2026**. O REA muda rápido: se algo estiver diferente, vale a versão da documentação oficial.

- Repositório oficial (morluto/rea) · site rea.tools · guias ilustrados · casos reais
- Instalação e recuperação
- Licença MIT · Aviso de responsabilidade

> Baixe o REA só pelas fontes oficiais acima (GitHub morluto/rea, rea.tools ou o pacote npm rea-agents). Repositórios com nome parecido não são oficiais.

Seu próximo passo: escolha uma funcionalidade pequena hoje e faça o ciclo completo — investigar → construir → testar.