

VIU — Documentação do MCP do cliente

Oct 14, 2026

Visão geral

O MCP do cliente VIU (servidor `viu-catalogo`, versão 1.0.0) dá ao Claude acesso ao catálogo VIU em quatro acervos: componentes de UI, estilos web, ícones e fontes. Ele oferece 12 ferramentas e 3 roteiros de trabalho (prompts), e está disponível para contas do plano Pro.

Acervo	O que o Claude recebe	Ferramentas
Componentes de UI	Nome, descrição, tags, prévia, dependências e o prompt completo com código e exemplos de uso; versão Kotlin / Compose Multiplatform quando existe	<code>catalog_status</code> , <code>search_components</code> , <code>get_component</code> , <code>list_tags</code>
Estilos web	Design systems de sites reais: paleta, tipografia, espaçamento e componentes, em DESIGN.md, Tailwind v4, variáveis CSS ou tokens DTCG	<code>search_styles</code> , <code>get_style</code> , <code>list_style_filters</code>
Ícones	SVG pronto (currentColor, 1em), licença, autor e componente React	<code>search_icons</code> , <code>get_icon</code> , <code>list_icon_collections</code>
Fontes	Pesos, eixos, licença, import CSS, download, pares usados em sites reais	<code>search_fonts</code> , <code>get_font</code>

O catálogo é atualizado continuamente. Por isso o Claude deve sempre buscar de novo em cada conversa, em vez de reaproveitar ids ou código que lembra.

O que o MCP do cliente não oferece: download em massa do acervo, gravação de arquivos no disco, kits de marca, banco de imagens, ferramentas de apps e qualquer informação sobre a origem dos componentes. Esses recursos ficam só no MCP da equipe.

Conexão e autenticação

O endereço do servidor é `https://api.virtues.now/vui/mcp`, com transporte Streamable HTTP e respostas JSON. O servidor só aceita POST e não abre stream SSE nem guarda sessão.

Forma de entrar	Como funciona	Quando usar
OAuth (login do VIU)	O cliente recebe 401 com <code>resource_metadata</code> , abre o login do VIU e obtém um token de acesso (<code>viuo_...</code> , vale 1 hora) e um de renovação (<code>viur_...</code> , vale 30 dias). Escopo: <code>mcp</code>	Conectores do claude.ai e outros clientes com OAuth
Chave da conta no cabeçalho	<code>Authorization: Bearer viu_...</code> ou <code>X-API-Key: viu_...</code> . A chave é gerada no perfil do site	Claude Code, Claude Desktop e clientes que mandam cabeçalho
Chave na URL	<code>https://api.virtues.now/vui/k/<chave>/mcp</code>	Clientes que não conseguem mandar cabeçalho

Exemplo no Claude Code:

```
claude mcp add --transport http viu https://api.virtues.now/vui/mcp --header "Authorization: Bearer viu_SUA_CHAVE"
```

Regras do protocolo:

- Uma mensagem JSON-RPC por requisição. Lotes (arrays) são recusados com erro `-32600`.
- Corpo de no máximo 1 MB.
- Métodos suportados: `initialize`, `ping`, `tools/list`, `tools/call`, `prompts/list`, `prompts/get` e notificações.
- A resposta de `initialize` traz as instruções de uso do servidor, que o Claude recebe automaticamente.
- A conta precisa estar no plano Pro. Outras contas recebem 403.

Regras de uso para o Claude

A regra central: buscar é grátis e abrir um componente gasta cota. O Claude escolhe pela busca e abre só o que vai usar.

- 1. Comece com `catalog_status` para ver o tamanho do catálogo e a cota restante.
- 2. Busque sempre nesta conversa. Não reaproveite ids, nomes ou código de conversas anteriores ou da memória: o catálogo muda o tempo todo.
- 3. Busque em inglês, pelo papel da peça (hero, navbar, pricing, testimonial, footer, cta, feature, form) e pelo estilo (glass, dark, gradient, 3d, minimal). Termos comuns em português são traduzidos automaticamente (rodapé → footer, preços → pricing, depoimentos → testimonial).
- 4. Prefira resultados com `code: true`. Use `has_code: true` na busca para filtrar.
- 5. Compare candidatos pela descrição, pelas tags e pela prévia (`preview`). Só então chame `get_component`.
- 6. Use o código de `get_component` como base e adapte ao projeto. Não reescreva de memória nem troque a peça por uma "parecida" inventada.
- 7. Instale as dependências listadas (npm e shadcn/ui) antes de usar o componente.
- 8. Para "no estilo de X": `search_styles`, depois `get_style`, e siga o DESIGN.md em tudo (cores, tipografia, espaçamento, componentes).
- 9. Para tela de app Android ou iOS: `search_components` com `mobile: true` e `get_component` com `platform: "kotlin"`.
- 10. Ícones: uma coleção só por página. Por padrão a busca exclui licenças não comerciais.
- 11. Fontes: todas as retornadas por padrão têm os acentos do português.

Para montar uma página ou seção inteira, o Claude pode seguir os roteiros do servidor:

Roteiro	Argumentos	O que faz
<code>montar_pagina</code>	<code>pedido</code> (obrigatório), <code>estilo</code> (opcional, ex.: Linear, Stripe)	Briefing → escolha das peças → montagem com o tema do projeto → revisão com nota em 1440, 768 e 390 px, até 3 rodadas
<code>montar_secao</code>	<code>secao</code> (obrigatório), <code>contexto</code> (opcional)	Uma seção (hero, planos, depoimentos...) num projeto existente, no tema dele
<code>revisar_pagina</code>	<code>onde</code> (opcional)	Revisão exigente com nota de 0 a 10 por critério e troca de peças pelo catálogo

No Claude Code os roteiros viram comandos, por exemplo `/mcp__viu__montar_pagina`. No claude.ai aparecem como prompts do conector.

Referência das ferramentas

Todas as ferramentas devolvem JSON como texto. Listas paginadas trazem `total_matches`, `offset` e `next_offset` (`null` na última página).

Componentes de UI

`catalog_status`

Tamanho do catálogo e cota do plano. Sem parâmetros. Não gasta cota de componentes.

Retorno: `plan`, `components` (`total`, `with_code`, `with_mobile_version`), `styles`, `icons`, `fonts` e `quota` (`components` com `limiteDia`, `limiteMes`, `usadosDia`, `usadosMes`, `restamDia`, `restamMes`; `calls`, `window_seconds`, `calls_in_window`).

`search_components`

Busca componentes por palavras-chave e filtros. Não gasta cota.

Parâmetro	Tipo	Padrão	Descrição
<code>query</code>	texto	—	Palavras-chave em inglês, ex.: <code>pricing table dark</code>
<code>tags</code>	lista de textos	—	Tags obrigatórias (todas precisam bater). Veja <code>list_tags</code>
<code>has_code</code>	booleano	<code>false</code>	Só componentes com código
<code>mobile</code>	booleano	<code>false</code>	Só componentes com versão Kotlin / Compose Multiplatform
<code>exclude_ids</code>	lista de textos	—	Ids para deixar de fora
<code>limit</code>	inteiro	10 (máx. 50)	Resultados por página
<code>offset</code>	inteiro	0	Início da página

Cada resultado: `id` (código público de 8 caracteres), `name`, `description` (até 160 caracteres), `tags` (até 8), `code`, `mobile` e `preview` (URL da prévia). O retorno também traz `query_terms`, os termos usados depois da tradução.

Ordenação: nome exato pesa mais, depois nome que começa com o termo, tag e descrição. Componentes com código e com prévia sobem.

```
{"query": "pricing dark", "has_code": true, "limit": 5}
```

get_component

O componente pronto para usar. Cada componente diferente aberto no dia conta uma vez na cota; abrir o mesmo de novo no mesmo dia não conta. Componente sem código não gasta cota.

Parâmetro	Tipo	Padrão	Descrição
<code>id</code>	texto (obrigatório)	—	Id vindo de <code>search_components</code>
<code>platform</code>	<code>web</code> ou <code>kotlin</code>	<code>web</code>	<code>kotlin</code> inclui a versão Kotlin / Compose Multiplatform, quando existe

Retorno: os campos da busca com a descrição completa, mais `dependencies` (npm e shadcn/ui), `prompt` (código e exemplos de uso, até 100.000 caracteres), `quota` atualizada e, quando houver, `video` e `mobile_hint`. Se o id não existir, volta `error` com até 5 componentes parecidos em `similar`.

```
{"id": "c1a1c9d4", "platform": "web"}
```

list_tags

Todas as tags com a contagem de componentes (`components`) e de componentes com código (`with_code`). Sem parâmetros.

Estilos web

search_styles

Busca design systems de sites reais. Sem `query`, lista todos, paginado.

Parâmetro	Tipo	Descrição
<code>query</code>	texto	Palavras-chave em inglês, ex.: <code>dark minimal fintech</code>
<code>industry</code>	texto	ex.: <code>saas, agency, ecommerce, ai, design</code>
<code>theme</code>	texto	claro, escuro ou misto (aceita <code>light, dark, mixed</code>)
<code>family</code>	texto	Família de cor, ex.: <code>neutro, azul</code>

Parâmetro	Tipo	Descrição
<code>collection</code>	texto	ex.: monocromatico, gradiente
<code>density</code>	texto	ex.: compact, comfortable
<code>font</code>	texto	Parte do nome da fonte, ex.: inter, serif
<code>limit</code> / <code>offset</code>	inteiro	10 (máx. 50) / 0

Cada resultado: `slug`, `name`, `site`, `theme`, `industry`, `north_star`, `fonts`, `colors` (até 6 hex), `accent`, `family`, `collections`, `density`.

get_style

Um estilo completo.

Parâmetro	Tipo	Padrão	Descrição
<code>slug</code>	texto (obrigatório)	—	Slug de <code>search_styles</code> ou o nome do estilo
<code>format</code>	<code>design_md</code> , <code>tailwind</code> , <code>css</code> , <code>tokens</code> , <code>all</code> , <code>meta</code>	<code>design_md</code>	Arquivo devolvido. <code>meta</code> não traz arquivo

Retorno: o resumo do estilo, `description`, `palette` (hex, nome, papel e grupo de cada cor), o arquivo pedido, `formats_available` e `media` (URLs de miniatura, tela, capa, vídeo e ícone). Se o slug não existir, volta `did_you_mean`.

list_style_filters

Valores válidos para os filtros de `search_styles`, com contagem: `industries`, `themes`, `families`, `densities`, `collections` e `fonts_top` (as 40 fontes mais usadas). Sem parâmetros.

Ícones

search_icons

Busca ícones em coleções open source (Lucide, Tabler, Phosphor, Material, Heroicons, marcas). Aceita português ou inglês.

Parâmetro	Tipo	Padrão	Descrição
<code>query</code>	texto	—	ex.: <code>shopping cart</code> , <code>seta esquerda</code> , <code>github</code>
<code>collection</code>	texto	—	Prefixo da coleção, ex.: <code>lucide</code> , <code>tabler</code> , <code>ph</code> , <code>mdi</code> , <code>simple-icons</code>
<code>include_noncommercial</code>	booleano	<code>false</code>	Inclui licenças não comerciais
<code>with_svg</code>	booleano	<code>true</code>	Traz o SVG em cada resultado
<code>limit / offset</code>	inteiro	20 (máx. 100) / 0	Paginação

Cada resultado: `id` (`prefixo:nome`), `name`, `collection`, `license` e `svg`. O SVG usa `currentColor` e `1em`, então herda a cor e o tamanho do texto.

get_icon

Um ícone pelo id.

Parâmetro	Tipo	Padrão	Descrição
<code>id</code>	texto (obrigatório)	—	<code>prefixo:nome</code> , ex.: <code>lucide:camera</code>
<code>format</code>	<code>svg</code> , <code>jsx</code> , <code>all</code>	<code>svg</code>	<code>jsx</code> ou <code>all</code> incluem um componente React pronto

Retorno: `svg`, `license`, `license_title`, `license_url`, `author`, `author_url` e, quando a licença pede cuidado, `warning`: não comercial, GPL ou atribuição obrigatória (CC-BY).

list_icon_collections

As coleções com `prefix`, `name`, `icons` (quantidade), `license` e `category`. Parâmetros: `query` (filtra por prefixo, nome ou categoria) e `limit` (padrão 60, máx. 300).

Fontes

search_fonts

Busca fontes do Google Fonts, Fontsource e Fontshare.

Parâmetro	Tipo	Padrão	Descrição
<code>query</code>	texto	—	Nome ou descrição, ex.: <code>serifada</code> <code>elegante</code> , <code>mono</code> , <code>inter</code>
<code>category</code>	<code>sans-serif</code> , <code>serif</code> , <code>display</code> , <code>handwriting</code> , <code>monospace</code>	—	Categoria
<code>source</code>	<code>google</code> , <code>fontsource</code> , <code>fontshare</code>	—	Origem da fonte
<code>variable_only</code>	booleano	false	Só fontes variáveis
<code>order</code>	texto	<code>popular</code>	popular, nome ou recente
<code>include_without_pt_accents</code>	booleano	false	Inclui fontes sem os acentos do português
<code>limit</code> / <code>offset</code>	inteiro	12 (máx. 60) / 0	Paginação

Cada resultado: `id`, `family`, `source`, `category`, `weights`, `italic`, `variable`, `license`, `can_host_files`, `css_import` e `sample` (imagem de amostra). Se nada bater, volta as mais populares da categoria com uma `note`.

get_font

Uma fonte completa. Parâmetros: `id` (de `search_fonts`) ou `family` (o nome).

Retorno: o resumo da busca, mais `designers`, `axes`, `description`, `tags`, `license_url`, `page`, `pt_accents`, `css` (import e font-family prontos), `download` (zip quando a licença permite hospedar), `pairs_in_real_sites` (fontes que aparecem junto em sites reais) e `seen_on_styles` (estilos web do catálogo que a usam).

Fluxos de exemplo

Landing page "no estilo de" um site

Pedido: "Monte a landing de uma fintech no estilo do Stripe."

1. `catalog_status` — confere a cota.
2. `search_styles` com `{"query": "stripe"}` e `get_style` no melhor resultado. O `DESIGN.md` define paleta, tipografia e espaçamento.
3. `search_fonts` / `get_font` para o par de fontes indicado no `DESIGN.md`.
4. Para cada seção, duas buscas: `search_components` com `{"query": "hero", "has_code": true}` e outra com as palavras do briefing (`"fintech gradient hero"`).
5. Monta uma tabela: seção, id, nome, motivo. Só então chama `get_component` em cada id escolhido.
6. `search_icons` com `{"collection": "lucide"}` para manter uma coleção só.
7. Adapta cada peça ao tema, instala as dependências, roda build e lint, e revisa em 1440, 768 e 390 px.

Uma seção num projeto existente

Pedido: "Adicione uma seção de preços."

1. Lê o tema do projeto (cores, fontes, raios).
2. `search_components` com `{"query": "preços", "has_code": true, "limit": 5}`. `"preços"` vira `pricing`.
3. Compara 3 a 5 candidatos pela descrição e pela prévia e abre só o escolhido com `get_component`.
4. Adapta ao tema e confere em 1440 e 390 px.

Tela de app mobile

Pedido: "Preciso de uma tela de login para o app Android."

1. `search_components` com `{"query": "login", "mobile": true}`.
2. `get_component` com `{"id": "<id>", "platform": "kotlin"}`. O prompt traz a versão web e a seção "Mobile — Kotlin" com o código Compose Multiplatform.
3. Adapta o pacote e o tema ao app.

Limites, identificadores e erros

Limites do plano Pro

Limite	Valor	Como conta
Chamadas	600 a cada 4 horas	Qualquer ferramenta, janela deslizante
Componentes por dia	150	Componentes diferentes abertos com <code>get_component</code> ; repetir o mesmo no dia não conta
Componentes por mês	1.500	Mesma regra, somada no mês

Buscas, listas, estilos, ícones e fontes gastam só chamadas, nunca a cota de componentes.

Identificadores

Acervo	Formato do id	Exemplo
Componente	Código público de 8 caracteres	<code>c1a1c9d4</code>
Estilo web	Slug	o <code>slug</code> devolvido por <code>search_styles</code>
Ícone	<code>prefixo:nome</code>	<code>lucide:camera</code>
Fonte	Id em minúsculas com hífens, ou o nome da família	o <code>id</code> devolvido por <code>search_fonts</code>

O id de componente só vale se veio de `search_components` . Ids em outro formato devolvem "id não encontrado" com sugestões parecidas. Prévias ficam em `https://api.virtues.now/vui/previews/p/<id>` .

Erros

Situação	Resposta	O que fazer
Sem credencial	HTTP 401 com desafio OAuth	Entrar pelo login do VIU ou mandar a chave

Situação	Resposta	O que fazer
Token OAuth vencido	HTTP 401 <code>invalid_token</code>	O cliente renova com o token de renovação
Chave inválida ou revogada	HTTP 401	Gerar nova chave no perfil do site
Conta fora do plano Pro	HTTP 403	Assinar o Pro
Limite de chamadas	HTTP 429 com <code>Retry-After</code> em segundos	Esperar o tempo indicado
Cota de componentes do dia ou do mês	Erro da ferramenta (<code>isError</code>): "Limite de 150 componentes por dia atingido. Volte amanhã."	Continuar só com buscas ou esperar o dia seguinte
Lote JSON-RPC	HTTP 400, código <code>-32600</code>	Mandar uma mensagem por requisição
JSON malformado	HTTP 400, código <code>-32700</code>	Corrigir o corpo
Método desconhecido	Código <code>-32601</code>	Usar um dos métodos suportados
GET ou DELETE	HTTP 405	Usar POST
Componente sem código	<code>code: false</code> e <code>note</code>	Escolher outro; não gasta cota

Erros dentro de uma ferramenta voltam ao Claude como resultado com `isError: true` e a mensagem, sem derrubar a conexão.