Web fetch tool

Copy page

The web fetch tool allows Claude to retrieve full content from specified web pages and PDF documents.

The web fetch tool is currently in beta. To enable it, use the beta header `web-fetch-2025-09-10` in your API requests.

Please use this form to provide feedback on the quality of the model responses, the API itself, or the quality of the documentation.

Enabling the web fetch tool in environments where Claude processes untrusted input alongside sensitive data poses data exfiltration risks. We recommend only using this tool in trusted environments or when handling non-sensitive data.

To minimize exfiltration risks, Claude is not allowed to dynamically construct URLs. Claude can only fetch URLs that have been explicitly provided by the user or that come from previous web search or web fetch results. However, there is still residual risk that should be carefully considered when using this tool.

If data exfiltration is a concern, consider:

- Disabling the web fetch tool entirely
- Using the `max_uses` parameter to limit the number of requests
- Using the `allowed_domains` parameter to restrict to known safe domains

Supported models

Web fetch is available on:

- Claude Sonnet 4.5 (`claude-sonnet-4-5-20250929`)
- Claude Sonnet 4 (`claude-sonnet-4-20250514`)
- Claude Sonnet 3.7 (deprecated) (`claude-3-7-sonnet-20250219`)

Ask Docs

- Claude Opus 4.5 (`claude-opus-4-5-20251101`)
- Claude Opus 4.1 (`claude-opus-4-1-20250805`)
- Claude Opus 4 (`claude-opus-4-20250514`)

How web fetch works

When you add the web fetch tool to your API request:

1. Claude decides when to fetch content based on the prompt and available URLs.
2. The API retrieves the full text content from the specified URL.
3. For PDFs, automatic text extraction is performed.
4. Claude analyzes the fetched content and provides a response with optional citations.

> The web fetch tool currently does not support web sites dynamically rendered via Javascript.

How to use web fetch

Provide the web fetch tool in your API request:

Shell

Ask Docs

```
    --header "anthropic-version: 2023-06-01" \
    --header "anthropic-beta: web-fetch-2025-09-10" \
    --header "content-type: application/json" \
    --data '{
        "model": "claude-sonnet-4-5",
        "max_tokens": 1024,
        "messages": [
            {
                "role": "user",
                "content": "Please analyze the content at https://example.com/
            }
        ],
        "tools": [{
            "type": "web_fetch_20250910",
            "name": "web_fetch",
            "max_uses": 5
        }]
    }'
```

Tool definition

The web fetch tool supports the following parameters:

JSON

```
  "name": "web_fetch",

  // Optional: Limit the number of fetches per request
  "max_uses": 10,

  // Optional: Only fetch from these domains
  "allowed_domains": ["example.com", "docs.example.com"],

  // Optional: Never fetch from these domains
  "blocked_domains": ["private.example.com"],

  // Optional: Enable citations for fetched content
  "citations": {
    "enabled": true
  },

  // Optional: Maximum content length in tokens
  "max_content_tokens": 100000
}
```

Max uses

The `max_uses` parameter limits the number of web fetches performed. If Claude attempts more fetches than allowed, the `web_fetch_tool_result` will be an error with the `max_uses_exceeded` error code. There is currently no default limit.

Domain filtering

When using domain filters:

- Domains should not include the HTTP/HTTPS scheme (use `example.com` instead of `https://example.com`)
- Subdomains are automatically included (`example.com` covers `docs.example.com`)
- Subpaths are supported (`example.com/blog`)
- You can use either `allowed_domains` or `blocked_domains`, but not both in the same request.

> ⚠ Be aware that Unicode characters in domain names can create security vulnerabilities through homograph attacks, where visually similar characters from different scripts can bypass domain filters. For example, `amazon.com` (using Cyrillic 'a') may appear identical to `amazon.com` but represents a different domain.

- Consider that URL parsers may handle Unicode normalization differently
- Test your domain filters with potential homograph variations
- Regularly audit your domain configurations for suspicious Unicode characters

Content limits

The `max_content_tokens` parameter limits the amount of content that will be included in the context. If the fetched content exceeds this limit, it will be truncated. This helps control token usage when fetching large documents.

> The `max_content_tokens` parameter limit is approximate. The actual number of input tokens used can vary by a small amount.

Citations

Unlike web search where citations are always enabled, citations are optional for web fetch. Set `"citations": {"enabled": true}` to enable Claude to cite specific passages from fetched documents.

> When displaying API outputs directly to end users, citations must be included to the original source. If you are making modifications to API outputs, including by reprocessing and/or combining them with your own material before displaying them to end users, display citations as appropriate based on consultation with your legal team.

Response

Here's an example response structure:

Claude Docs

Ask Docs

```
// 1. Claude's decision to fetch
{
  "type": "text",
  "text": "I'll fetch the content from the article to analyze it."
},
// 2. The fetch request
{
  "type": "server_tool_use",
  "id": "srvtoolu_01234567890abcdef",
  "name": "web_fetch",
  "input": {
    "url": "https://example.com/article"
  }
},
// 3. Fetch results
{
  "type": "web_fetch_tool_result",
  "tool_use_id": "srvtoolu_01234567890abcdef",
  "content": {
    "type": "web_fetch_result",
    "url": "https://example.com/article",
    "content": {
      "type": "document",
      "source": {
        "type": "text",
        "media_type": "text/plain",
        "data": "Full text content of the article..."
      },
      "title": "Article Title",
      "citations": {"enabled": true}
    },
    "retrieved_at": "2025-08-25T10:30:00Z"
  }
},
// 4. Claude's analysis with citations (if enabled)
{
  "text": "Based on the article, ",
  "type": "text"
},
{
  "text": "the main argument presented is that artificial intelligence wil
  "type": "text",
  "citations": [
    {
      "type": "char_location",
      "document_index": 0,
      "document_title": "Article Title",
```

```
          "cited_text": "Artificial intelligence is poised to revolutionize he
        }
      ]
    }
  ],
  "id": "msg_a930390d3a",
  "usage": {
    "input_tokens": 25039,
    "output_tokens": 931,
    "server_tool_use": {
      "web_fetch_requests": 1
    }
  },
  "stop_reason": "end_turn"
}
```


Fetch results

Fetch results include:

- `url` : The URL that was fetched
- `content` : A document block containing the fetched content
- `retrieved_at` : Timestamp when the content was retrieved

> ⓘ The web fetch tool caches results to improve performance and reduce redundant requests. This means the content returned may not always be the latest version available at the URL. The cache behavior is managed automatically and may change over time to optimize for different content types and usage patterns.

For PDF documents, the content will be returned as base64-encoded data:

```
  "tool_use_id": "srvtoolu_02",
  "content": {
    "type": "web_fetch_result",
    "url": "https://example.com/paper.pdf",
    "content": {
      "type": "document",
      "source": {
        "type": "base64",
        "media_type": "application/pdf",
        "data": "JVBERi0xLjQKJcOkw7zDtsOfCjIgMCBvYmo..."
      },
      "citations": {"enabled": true}
    },
    "retrieved_at": "2025-08-25T10:30:02Z"
  }
}
```

Errors

When the web fetch tool encounters an error, the Claude API returns a 200 (success) response with the error represented in the response body:

```
{
  "type": "web_fetch_tool_result",
  "tool_use_id": "srvtoolu_a93jad",
  "content": {
    "type": "web_fetch_tool_error",
    "error_code": "url_not_accessible"
  }
}
```

These are the possible error codes:

- `invalid_input` : Invalid URL format
- `url_too_long` : URL exceeds maximum length (250 characters)
- `url_not_allowed` : URL blocked by domain filtering rules and model restrictions
- `url_not_accessible` : Failed to fetch content (HTTP error)
- `too_many_requests` : Rate limit exceeded
- `unsupported_content_type` : Content type not supported (only text and PDF)
- `max_uses_exceeded` : Maximum web fetch tool uses exceeded

URL validation

For security reasons, the web fetch tool can only fetch URLs that have previously appeared in the conversation context. This includes:

- URLs in user messages
- URLs in client-side tool results
- URLs from previous web search or web fetch results

The tool cannot fetch arbitrary URLs that Claude generates or URLs from container-based server tools (Code Execution, Bash, etc.).

Combined search and fetch

Web fetch works seamlessly with web search for comprehensive information gathering:

```python
client = anthropic.Anthropic()

response = client.messages.create(
    model="claude-sonnet-4-5",
    max_tokens=4096,
    messages=[
        {
            "role": "user",
            "content": "Find recent articles about quantum computing and analy
        }
    ],
    tools=[
        {
            "type": "web_search_20250305",
            "name": "web_search",
            "max_uses": 3
        },
        {
            "type": "web_fetch_20250910",
            "name": "web_fetch",
            "max_uses": 5,
            "citations": {"enabled": True}
        }
    ],
    extra_headers={
        "anthropic-beta": "web-fetch-2025-09-10"
    }
)
```

In this workflow, Claude will:

1. Use web search to find relevant articles
2. Select the most promising results
3. Use web fetch to retrieve full content
4. Provide detailed analysis with citations

Prompt caching

Web fetch works with prompt caching. To enable prompt caching, add `cache_control` breakpoints in your request. Cached fetch results can be reused across conversation turns.

Claude Docs

Ask Docs

```
# First request with web fetch
messages = [
    {
        "role": "user",
        "content": "Analyze this research paper: https://arxiv.org/abs/2024.12
    }
]

response1 = client.messages.create(
    model="claude-sonnet-4-5",
    max_tokens=1024,
    messages=messages,
    tools=[{
        "type": "web_fetch_20250910",
        "name": "web_fetch"
    }],
    extra_headers={
        "anthropic-beta": "web-fetch-2025-09-10"
    }
)

# Add Claude's response to conversation
messages.append({
    "role": "assistant",
    "content": response1.content
})

# Second request with cache breakpoint
messages.append({
    "role": "user",
    "content": "What methodology does the paper use?",
    "cache_control": {"type": "ephemeral"}
})

response2 = client.messages.create(
    model="claude-sonnet-4-5",
    max_tokens=1024,
    messages=messages,
    tools=[{
        "type": "web_fetch_20250910",
        "name": "web_fetch"
    }],
    extra_headers={
        "anthropic-beta": "web-fetch-2025-09-10"
    }
)
```

```
print(f"Cache read tokens: {response2.usage.get('cache_read_input_tokens', 0)}")
```

Streaming

With streaming enabled, fetch events are part of the stream with a pause during content retrieval:

```
event: message_start
data: {"type": "message_start", "message": {"id": "msg_abc123", "type": "messa

event: content_block_start
data: {"type": "content_block_start", "index": 0, "content_block": {"type": "t

// Claude's decision to fetch

event: content_block_start
data: {"type": "content_block_start", "index": 1, "content_block": {"type": "s

// Fetch URL streamed
event: content_block_delta
data: {"type": "content_block_delta", "index": 1, "delta": {"type": "input_jso

// Pause while fetch executes

// Fetch results streamed
event: content_block_start
data: {"type": "content_block_start", "index": 2, "content_block": {"type": "w

// Claude's response continues...
```

Batch requests

You can include the web fetch tool in the Messages Batches API. Web fetch tool calls through the Messages Batches API are priced the same as those in regular Messages API requests.

Usage and pricing

Web fetch usage has **no additional charges** beyond standard token costs:

```
  "output_tokens": 931,
  "cache_read_input_tokens": 0,
  "cache_creation_input_tokens": 0,
  "server_tool_use": {
    "web_fetch_requests": 1
  }
}
```


The web fetch tool is available on the Claude API at **no additional cost**. You only pay standard token costs for the fetched content that becomes part of your conversation context.

To protect against inadvertently fetching large content that would consume excessive tokens, use the `max_content_tokens` parameter to set appropriate limits based on your use case and budget considerations.

Example token usage for typical content:

- Average web page (10KB): ~2,500 tokens
- Large documentation page (100KB): ~25,000 tokens
- Research paper PDF (500KB): ~125,000 tokens

Claude Docs

Google Cloud's Vertex AI

Company

Anthropic

Careers

Economic Futures

Research

News

Responsible Scaling Policy

Security and compliance

Transparency

Help and security

Availability

Status

Support

Discord

Terms and policies

Privacy policy

Responsible disclosure policy

Terms of service: Commercial

Terms of service: Consumer

Usage policy

Ask Docs