

 \equiv

Scrape > Batch Scrape

Scrape

Batch Scrape

Batch scrape multiple URLs

Batch scraping multiple URLs

You can now batch scrape multiple URLs at the same time. It takes the starting URLs and optional parameters as arguments. The params argument allows you to specify additional options for the batch scrape job, such as the output formats.

How it works

It is very similar to how the <code>/crawl</code> endpoint works. It submits a batch scrape job and returns a job ID to check the status of the batch scrape.

The sdk provides 2 methods, synchronous and asynchronous. The synchronous method will return the results of the batch scrape job, while the asynchronous method will return a job ID that you can use to check the status of the batch scrape.

Usage

from firecrawl import FirecrawlApp

app = FirecrawlApp(api_key="fc-YOUR_API_KEY")

Scrape multiple websites:
batch_scrape_result = app.batch_scrape_urls(['firecrawl.dev', 'mendable.ai'], {'

```
print(batch_scrape_result)
    Firecrawl
```

```
# Or, you can use the asynchronous method:
batch_scrape_ioh = app.async_batch_scrape_urls(['firecrawl.dev', 'mendable.ai'],
Scrape > Batch Scrape
print(batch_scrape_job)
```

```
# (async) You can then use the job ID to check the status of the batch scrape:
batch_scrape_status = app.check_batch_scrape_status(batch_scrape_job['id'])
print(batch_scrape_status)
```

Response

If you're using the sync methods from the SDKs, it will return the results of the batch scrape job. Otherwise, it will return a job ID that you can use to check the status of the batch scrape.

Synchronous

Completed

```
{
  "status": "completed",
  "total": 36,
  "completed": 36,
  "creditsUsed": 36,
  "expiresAt": "2024-00-00T00:00:00.000Z",
  "next": "https://api.firecrawl.dev/v1/batch/scrape/123-456-789?skip=26",
  "data": [
    {
      "markdown": "[Firecrawl Docs home page![light logo](https://mintlify.s3-us-
      "html": "<!DOCTYPE html><html lang=\"en\" class=\"js-focus-visible lg:[--sc
      "metadata": {
        "title": "Build a 'Chat with website' using Grog Llama 3 | Firecrawl",
        "language": "en",
        "sourceURL": "https://docs.firecrawl.dev/learn/rag-llama3",
        "description": "Learn how to use Firecrawl, Grog Llama 3, and Langchain to
        "oqLocaleAlternate": [],
```

```
"statusCode": 200

Firegrawl

Scrape > Batch Scrape
```

Asynchronous

You can then use the job ID to check the status of the batch scrape by calling the /batch/scrape/{id} endpoint. This endpoint is meant to be used while the job is still running or right after it has completed as batch scrape jobs expire after 24 hours.

```
{
    "success": true,
    "id": "123-456-789",
    "url": "https://api.firecrawl.dev/v1/batch/scrape/123-456-789"
}
```

Batch scrape with extraction

cURL

You can also use the batch scrape endpoint to extract structured data from the pages. This is useful if you want to get the same structured data from a list of URLs.

```
from firecrawl import FirecrawlApp

app = FirecrawlApp(api_key="fc-YOUR_API_KEY")

# Scrape multiple websites:
batch_scrape_result = app.batch_scrape_urls(
    ['https://docs.firecrawl.dev', 'https://docs.firecrawl.dev/sdks/overview'],
    {
        'formats': ['extract'],
```

Python

Node

```
'extract': {
Firecrawl 'prompt': 'Extract the title and description from the page.',
            'schema': {
 Scrape > Batch Scrape ': 'object',
                'properties': {
                    'title': {'type': 'string'},
                    'description': {'type': 'string'}
                },
                'required': ['title', 'description']
            }
        }
    }
)
print(batch_scrape_result)
# Or, you can use the asynchronous method:
batch_scrape_job = app.async_batch_scrape_urls(
    ['https://docs.firecrawl.dev', 'https://docs.firecrawl.dev/sdks/overview'],
    {
        'formats': ['extract'],
        'extract': {
            'prompt': 'Extract the title and description from the page.',
            'schema': {
                'type': 'object',
                'properties': {
                    'title': {'type': 'string'},
                    'description': {'type': 'string'}
                },
                'required': ['title', 'description']
            }
        }
    }
print(batch_scrape_job)
# (async) You can then use the job ID to check the status of the batch scrape:
batch_scrape_status = app.check_batch_scrape_status(batch_scrape_job['id'])
print(batch_scrape_status)
```



Scrape > Batch Scrape

Completed

Asynchronous

```
{
    "success": true,
    "id": "123-456-789",
    "url": "https://api.firecrawl.dev/v1/batch/scrape/123-456-789"
}
```



Scrape > Batch Scrape < Scrape

LLM Extract >

Powered by Mintlify