

API:Main page

 This page is part of the MediaWiki action API documentation.


This is an overview of the "action" API. See the menu bar on the right for more detailed sub-topics and other APIs.


The MediaWiki action API is a web service that provides convenient access to wiki features, data, and meta-data over HTTP, via a URL usually at `api.php`. Clients request particular "actions" by specifying an `action` parameter, mainly `action=query` to get information. It was known as *the* MediaWiki API, but there are now other web APIs available that connect to MediaWiki such as RESTBase and the Wikidata query service.

Contents

- 1 Introduction
- 2 A simple example
 - 2.1 The endpoint
 - 2.2 The format
 - 2.3 The action
 - 2.4 Action-specific parameters
- 3 Getting started
- 4 Identifying your client
- 5 Logging in
- 6 API etiquette
- 7 Useful links
- 8 Archived links

Introduction

 **Note:** If you are instead looking for an "internal API" or "PHP API", see the **extension interface**, which allows PHP developers to add new functionality to a MediaWiki installation.

 Deprecation notices are sent to the low-traffic **mediawiki-api-announce** mailing list. Subscription is recommended.

The MediaWiki action API can be used to monitor a MediaWiki installation, or create a bot to automatically maintain one. It provides direct, high-level access to the data contained in MediaWiki databases. Client programs can log in to a wiki, get data, and post changes automatically by making HTTP requests to the web service. Supported clients include bots, thin web-based JavaScript clients such as Navigation popups and LiveRC, end-user applications such as Vandal Fighter, and other web sites (Toolforge (<https://tools.wmflabs.org>)'s utilities).

On new MediaWiki installations, the web service is enabled by default, but an administrator can disable it.

MediaWiki has two other outward-facing interfaces:

- The **Special:Export** page, which provides bulk export of wiki content as XML. Read Help:Export for more information.
- The **standard web-based interface** (which you are likely using right now to view this page). Read Manual:Parameters to index.php for information on using the web-based interface.

A simple example

This URL tells English Wikipedia's web service API to send you the content of the main page:

`https://en.wikipedia.org/w/api.php?action=query&titles=Main%20Page&prop=revisions&rvprop=content&format=json`

Use any programming language to make an HTTP GET request for that URL (or just visit that link in your browser), and you'll get a JSON document which includes the current wiki markup for the page titled "Main Page". Changing the format to `jsonfm` will return a "pretty-printed" HTML result good for debugging.

Here is the `jsonfm` URL as an easier-to-read clickable link.

api.php ? action=query & titles=Main%20Page & prop=revisions & rvprop=content & format=jsonfm (<https://en.wikipedia.org/w/api.php?action=query&titles=Main%20Page&prop=revisions&rvprop=content&format=jsonfm>) [try in ApiSandbox] (<https://en.wikipedia.org/wiki/Special:ApiSandbox#action=query&titles=Main%20Page&prop=revisions&rvprop=content&format=jsonfm>)

MediaWiki APIs

Search API pages

- Web APIs showcase
- REST content API (<https://www.mediawiki.org/api/>)

MediaWiki action API

- Introduction and quick start**
- FAQ
 - Etiquette and usage limits
- Tutorial
- Formats
- Error reporting
- Restricting usage
- Cross-site requests
- Authentication
 - Login
 - Logout
 - Account creation
- Queries
 - Meta information
 - Properties
 - Lists
- Searching (by title, content, coordinates...)
- Parsing wikitext and expanding templates
- Purging pages' caches
- Parameter information
- Changing wiki content
 - Create and edit pages
 - Move pages
 - Merge pages
 - Rollback
 - Delete pages
 - Restore deleted revisions
 - (Un)protect pages
 - (Un)block users
 - (Un)watch pages
 - Mark revisions of watched pages as visited
 - Send email
 - Patrol changes
 - Import pages
 - Change user group membership
 - Upload files
 - User options
 - Tokens
 - Page language
 - More...
- Watchlist feed
- Wikidata
- Extensions
- Using the API in MediaWiki and extensions
- Miscellaneous
- Implementation
 - Localisation
- Client code
- Asserting

v · d · e (<https://www.mediawiki.org/w/index.php?title=Template:API&action=edit>)

Let's pick that URL apart to show how it works.

The endpoint

https://en.wikipedia.org/w/api.php

This is the *endpoint*. It's like the home page of the MediaWiki web service API. This URL is the base URL for English Wikipedia's API, just as https://en.wikipedia.org/wiki/ is the base URL for its web site.

If you're writing a program to use English Wikipedia, every URL you construct will begin with this base URL. If you're using a different MediaWiki installation, you'll need to find its endpoint and use that instead. All Wikimedia wikis have endpoints that follow this pattern:

```
https://www.mediawiki.org/w/api.php      # MediaWiki API
https://en.wikipedia.org/w/api.php      # English Wikipedia API
https://nl.wikipedia.org/w/api.php      # Dutch Wikipedia API
https://commons.wikimedia.org/w/api.php # Wikimedia Commons API
```

Since r75621, we have RSD discovery for the endpoint: look for the link `rel="EditURI"` in the HTML source of any page and extract the `api.php` URL; the actual link contains additional info. For instance, on this wiki it's:

MediaWiki version:

≥ 1.17

```
<link rel="EditURI" type="application/rsd+xml" href="//www.mediawiki.org/w/api.php?action=rsd" />
```

Otherwise, there's no safe way to locate the endpoint on any wiki. If you're lucky, either the full path to `index.php` will not be hidden under strange rewrite rules so that you'll only have to take the "edit" (or history) link and replace `index.php` (etc.) with `api.php`, or you'll be able to use the default script path (like `w/api.php`).

Now let's move on to the parameters in the query string of the URL.

The format

`format=json` This tells the API that we want data to be returned in JSON format. You might also want to try `format=jsonfm` to get an HTML version of the result that is good for debugging. The API supports other output formats such as XML and native PHP (<http://php.net/manual/en/function.serialize.php>), but there are plans to remove less popular formats ([phab:T95715](#)), so you might not want to use them.

The action

`action=query`

The MediaWiki web service API implements dozens of actions and extensions implement many more; the dynamically generated API help (<https://en.wikipedia.org/w/api.php>) documents all available actions on a wiki. In this case, we're using the "query" action to get some information. The "query" action is one of the API's most important actions, and it has extensive documentation of its own. What follows is just an explanation of a single example.

Action-specific parameters

`titles=Main%20Page`

The rest of the example URL contains parameters used by the "query" action. Here, we're telling the web service API that we want information about the Wiki page called "Main Page". (The `%20` comes from percent-encoding a space.) If you need to query multiple pages, put them all in one request to optimize network and server resources: `titles=PageA|PageB|PageC`. See the query documentation for details.

`prop=revisions`

You can request many kinds of information, or **properties**, about a page. This parameter tells the web service API that we want information about a particular revision of the page. Since we're not specifying any revision information, the API will give us information about the latest revision — the main page of Wikipedia as it stands right now.

`rvprop=content`

Finally, this parameter tells the web service API that we want the content of the latest revision of the page. If we passed in `rvprop=content|user` instead, we'd get the latest page content *and* the name of the user who made the most recent revision.

Again, this is just one example. Queries are explained in more detail [here](#), and the API reference (<https://en.wikipedia.org/w/api.php>) lists all the possible actions, all the possible values for `rvprop`, and so on.

Getting started

Before you start using the MediaWiki web service API, be sure to read these documents:

- The FAQ.
- The page about input and output formats
- The page about errors and warnings
- Any policies that apply to the wiki you want to access, such as Wikimedia Foundation wikis' terms of use, trademark policy. These terms apply to you when you access or edit using the API, just as they do when you use your web browser.

Beyond that point, what you need to read depends on what you want to do. The right-hand menu links to detailed, task-specific documentation, and some more general guidelines are given below.

Identifying your client

When you make HTTP requests to the MediaWiki web service API, be sure to specify a User-Agent header that properly identifies your client. Don't use the default User-Agent provided by your client library, but make up a custom header that identifies your script or service and provides some type of means of contacting you (e.g., an e-mail address).

An example User-Agent string might look like:

```
MyCoolTool/1.1 (https://example.org/MyCoolTool/; MyCoolTool@example.org) BasedOnSuperLib/1.4
```

On Wikimedia wikis, if you don't supply a User-Agent header, or you supply an empty or generic one, your request will fail with an HTTP 403 error (cf. m:User-Agent policy). Other MediaWiki installations may have similar policies.

If you are calling the API from browser-based JavaScript, you won't be able to influence the User-Agent header: the browser will use its own. To work around this, use the Api-User-Agent header:

```
// Using XMLHttpRequest
xhr.setRequestHeader( 'Api-User-Agent', 'Example/1.0' );

// Using jQuery
$.ajax( {
  url: remoteUrlWithOrigin,
  data: queryData,
  dataType: 'json',
  type: 'POST',
  headers: { 'Api-User-Agent': 'Example/1.0' },
  success: function(data) {
    // do something with data
  }
} );

// Using mw.Api, specify it when creating the mw.Api object
var api = new mw.Api( {
  ajax: {
    headers: { 'Api-User-Agent': 'Example/1.0' }
  }
} );
api.get( {...} ).done(function(data) {
  // do something with data
});
```

In PHP, you can identify your user-agent with code such as this:

```
ini_set('user_agent', 'MyCoolTool/1.1 (https://example.org/MyCoolTool/; MyCoolTool@example.org) BasedOnSuperLib/1.4');
```

Or if you use cURL:

```
curl_setopt($curl, CURLOPT_USERAGENT, 'MyCoolTool/1.1 (https://example.org/MyCoolTool/; MyCoolTool@example.org) BasedOnSuperLib/1.4');
```

Logging in

Your client will probably need to log in to MediaWiki, possibly via its own user account. See the login manual page for details.

API etiquette

Please also read: API:Etiquette

If your requests obtain data that can be cached for a while, you should take steps to cache it, so you don't request the same data over and over again. More information about rate-limiting, concurrency, and general API etiquette can be found at API:Etiquette. Some clients may be able to cache data themselves, but for others (particularly JavaScript clients), this is not possible.

Per the HTTP specification, POST requests cannot be cached. Therefore, whenever you're reading data from the web service API, you should use GET requests, not POST.

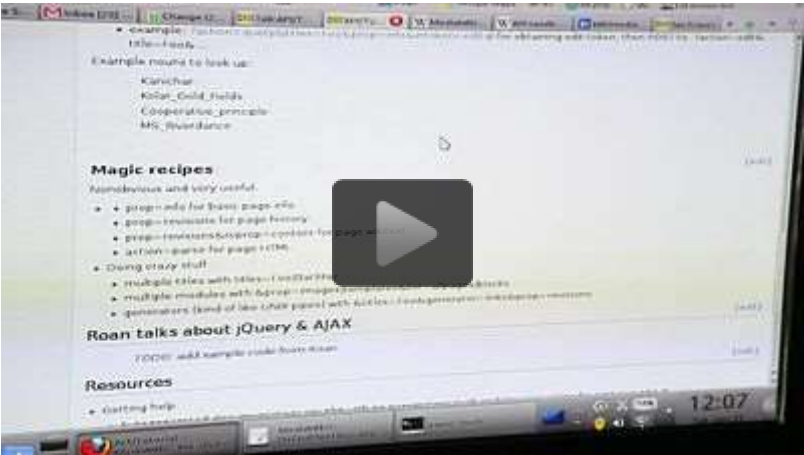
Also note that a request cannot be served from cache unless the URL is **exactly the same**. If you make a request for `api.php?...titles=Foo|Bar|Hello`, and cache the result, then a request for `api.php?...titles=Hello|Bar|Hello|Foo` will not go through the cache — even though MediaWiki returns the same data!

You should take care to normalize the URLs you send to the MediaWiki web service, so that slightly different user input won't cause you to waste time on unnecessary HTTP requests. You can normalize a list of page titles by **removing duplicates** and **sorting the titles alphabetically**. Similar techniques will work for other kinds of data.

Useful links

The menu bar on the right side of this page links to more detailed, task-specific documentation. Here are some links having to do with the API as a whole:

- The API sandbox available on all Wikimedia wikis makes it easy to try out different actions interactively.
- The API reference (<https://en.wikipedia.org/w/api.php>) contains automatically-generated descriptions of all actions and parameters.
- Hook into Wikipedia information using PHP and the MediaWiki API (<https://www.ibm.com/developerworks/xml/library/x-phpwikipedia/index.html>) (IBM developerWorks article, 17 May 2011)



An introduction to the API by Roan Kattouw at the San Francisco Hackathon January 2012

- 8/15/2017

API:Main page - MediaWiki
- Hook into Wikipedia using Java and the MediaWiki API (<http://www.integratingstuff.com/2012/04/06/hook-into-wikipedia-using-java-and-the-mediawiki-api/>) (6 April 2012)
 - The API tutorial leads you through hands-on exercises and includes a training video.
 - Mailing list for notifications and questions: **API mailing list**
 - Low-traffic mailing list for announcements only (all posts to this list are posted to mediawiki-api as well): **mediawiki-api-announce**
 - View and report API bugs in the **MediaWiki-API** Phabricator project (<https://phabricator.wikimedia.org/tag/mediawiki-api/>) (*When reporting new bugs (<https://phabricator.wikimedia.org/maniphest/task/create/?projects=MediaWiki-API>), don't forget to add MediaWiki-API to Projects*)
 - Browse the API source code
 - Manual:Database layout

asdasdasdsad — The current MediaWiki database schema

- Browse the current database schema in git

Archived links

- 2006 API discussion

Retrieved from "https://www.mediawiki.org/w/index.php?title=API:Main_page&oldid=2537050"

-
- This page was last edited on 14 August 2017, at 13:36.
 - Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. See Terms of Use for details.