mediawiki-api

Release 0.6

Oct 01, 2018

Contents

Welcome to the documentation for the `addwiki/mediawiki-api` package! This is part of the Addwiki family of PHP packages.

Quick links:

- This documentation: http://addwiki.readthedocs.io/projects/mediawiki-api/
- Source code: https://github.com/addwiki/mediawiki-api/
- Issue tracker: https://phabricator.wikimedia.org/project/profile/1490/

CHAPTER 1

Contents

1.1 Page Lists

The Page List Getter allows you to retrieve lists of pages based on various criteria. It takes care of continuing queries where they span multiple requests, ensuring that you get all pages in your result set. This means that for some lists of pages a great many requests will be sent, and you should account for this possible performance problem when you request these lists (e.g. by running these as a background process and caching the results).

To use it, first get a new PageListGetter object from the factory:

```
$api = new \Mediawiki\Api\MediawikiApi( 'http://localhost/w/api.php' );
$services = new \Mediawiki\Api\MediawikiFactory( $api );
$pageListGetter = $services->newPageListGetter();
```

The examples below all use this `$pageListGetter` object.

All methods of the PageListGetter return `Page` objects; this class is part of the addwiki/mediawiki-datamodel package, and is documented in that page's documentation.

1.1.1 All pages in a category

Note that the category name as provided should also include the 'Category' namespace prefix (in the language of the wiki, or in canonical English form).

```
$examplePages = $pageListGetter->getPageListFromCategoryName( 'Category:Example pages
↪' );
foreach ( $examplePages->asArray() as $exPage ) {
    echo $exPage->getTitle()->getText();
}
```

1.1.2 Pages that transclude a template

Although generally it is templates that are transcluded, any page may be and so any page title can be passed to this method.

```
$usingTestTemplate = $pageListGetter->getPageListFromPageTransclusions( 'Template:Test
↪' );
```

1.1.3 Pages that link to a given page

Get the list of pages that link to a particular page.

```
$backLinks = $pageListGetter->getFromWhatLinksHere( 'Test page' );
```

1.1.4 Pages with a given prefix

Find pages that have a particular prefix to their title. This can also be used to find subpages of any page.

```
$backLinks = $pageListGetter->getFromPrefix( 'A page/' );
```

1.1.5 Random pages

Get up to ten random pages at a time. This method takes the same arguments as the API list=random query.

- `rnlimit` How many pages to get. No more than 10 (20 for bots) allowed. Default: 1.
- `rnnamespace` Pipe-separate list of namespace IDs.
- `rnfilterredir` How to filter for redirects. Possible values: `all`, `redirects`, `nonredirects`. Default: `nonredirects`.

```
$backLinks = $pageListGetter->getRandom( [
    'rnlimit' => 7,
    'rnnamespace' => '3|5|6',
    'rnfilterredir' => 'all',
] );
```

1.2 Category Traversal

The CategoryTraverser class is used to start at one Category page in a wiki's category hierarchy and descend through that category's children, grandchildren, and so on. The basic output of this is a Pages object containing all the pages in the category tree. It is also possible to register callbacks that will be called for every subcategory or other page (i.e. anything not a category).

1.2.1 Basic usage

To get all pages in a category or any of its subcategories.

```php
// Construct the API.
$api = new \Mediawiki\Api\MediawikiApi( 'http://localhost/w/api.php' );
$services = new \Mediawiki\Api\MediawikiFactory( $api );
$categoryTraverser = $services->newCategoryTraverser();

// Get the root category.
$rootCatIdent = new PageIdentifier( new Title( 'Category:Categories' ) );
$rootCat = $this->factory->newPageGetter()->getFromPageIdentifier( $pageIdentifier );

// Get all pages.
$allPages = $categoryTraverser->descend( $rootCat );
```

1.3 Getting Namespaces

The Name Space Getter allows you to search for namespaces and their aliases and to list all namespaces of a wiki.

To use it, first get a new NamespaceGetter object from the factory:

```php
$api = new \Mediawiki\Api\MediawikiApi( 'http://localhost/w/api.php' );
$services = new \Mediawiki\Api\MediawikiFactory( $api );
$namespaceGetter = $services->newNamespaceGetter();
```

1.3.1 Looking for a namespace

If you've got a page name like `File:awesome_cats.jpg` and want to know its namespace ID and possible localized names and aliases, use the following code:

```php
$fileNamespace = $namespaceGetter->getNamespaceByName( 'File' );
printf( "Name in local language: %s\n", $fileNamespace->getLocalName() );
printf( "Possible aliases: %s\n", implode( ', ', $fileNamespace->getAliases() ) );
// ... etc
```

`getNamespaceByName` accepts the canonical name, the local name and aliases. If you want to match only the canonical name, use `getNamespaceByCanonicalName` instead.

1.3.2 Getting a namespaced page

If you have a page title that is not in the default namespace, you can't pass the page name string `PageGetter` but must construct a `Title` object instead:

```php
$pageName = 'User:MalReynolds';
$nameParts = explode( ':', $pageName, 2 );
$namespace = $namespaceGetter->getNamespaceByName( $nameParts[0] );
$title = new \Mediawiki\DataModel\Title( $nameParts[1], $namespace->getId() );
$page = $services->newPageGetter()->getFromTitle( $title );
```

1.3.3 Listing all namespaces

```
foreach( $namespaceGetter->getNamespaces() as $namespace ) {
   echo $namespace->getLocalName() .  "\n";
}
```

1.4 Uploading files

1.4.1 Basic usage

To upload a single, small-sized file:

```
// Construct the API.
$api = new \Mediawiki\Api\MediawikiApi( 'http://localhost/w/api.php' );
$services = new \Mediawiki\Api\MediawikiFactory( $api );
$fileUploader = $services->newFileUploader();

// Upload the file.
$fileUploader->upload( 'The_file.png', '/full/path/to/the_file.png' );
```

If you need to work with larger files, you can switch to chunked uploading:

```
// Upload the file in 10 MB chunks.
$fileUploader = $services->newFileUploader();
$fileUploader->setChunkSize( 1024 * 1024 * 10 );
$fileUploader->upload( 'The_file.png', '/full/path/to/the_file.png' );
```

1.5 Contributing

We welcome all contributions, be they code, documentation, or even just ideas about how to make this package better!

The best way to get started is to browse the **'#addwiki board on Phabricator'_** and either work on one of the tasks already there or create a new one with details of what you want to work on.

1.5.1 Get the code

The code is hosted on GitHub. Clone the repository with:

```
$ git clone https://github.com/addwiki/mediawiki-api.git
```

1.5.2 Run the tests

After cloning the repository and updating the dependencies with Composer, you should be able to run all **unit** tests with:

```
./vendor/bin/phpunit ./tests/unit
```

To run the **integration** tests you need to set up a local MediaWiki installation (including with a `admin` administrator user with password `admin123`) and tell `phpunit` where to find it.

1. Copy `./phpunit.xml.dist` to `./phpunit.xml` and add the following section:

```
<php>
    <env name="MEDIAWIKI_API_URL" value="http://localhost/path/to/your/wiki/api.
↪php" />
</php>
```

2. Create and promote a new user:

```
$ php mediawiki/maintenance/createAndPromote.php --sysop WikiSysop wiki123sysop
```

Now all integration tests can be run with:

```
./vendor/bin/phpunit ./tests/integration
```