WishList Member
Application Programming Interface 2.0

for WishList Member 2.61.1022 or higher

www.wishlistproducts.com

Table of Contents

Overview

The WishList Member Application Programming Interface (API) version 2.0 provides an easy way for third-party developers to create applications that interact with some of the functionalities of WishList Member.

If is this document's goal to provide developers with as much as information as possible about the API. Examples will be provided using the PHP scripting language.

What's Needed

In order to make the API work, a developer must have access to the following:

- A copy of WishList Member 2.61.1022 or later installed and activated on a WordPress install.
- cURL if using PHP or an equivalent library that allows transferring of HTTP data and cookie handling if using a different language
- Average programming skills

Technology Used

Talking to the API

Applications must make use of Representational State Transfer or REST to communicate with the WishList Member API. Any programming language that supports REST (which is pretty much everything) can be used.

The reason for choosing REST over other transfer protocols is its simplicity. One simply needs to construct a valid URL to call the API and send data using the regular HTTP methods GET, POST, PUT and DELETE.

Data Return Format

To make it simple, the WishList Member API returns data in three different and widely used formats, namely:

- Serialized PHP Data
- JSON
- XML

This makes it easy to write web applications and desktop applications alike.

System Requirements

This API requires WishList Member 2.61.1022 installed on a server running Apache, PHP 5.2.4 or higher, PHP cURL 7.19.7 or higher, and MySQL 5.0 or higher

REST Methods

The API makes use of four HTTP methods which we call verbs. Each method is equivalent to an action that is to be performed by the API. They are:

Method (Verb)	Equivalent Action
POST	Create
GET	Read
PUT	Update
DELETE	Delete

Quick Example Using GET

Let's get started.

The WishList Member API 2.0 can be accessed by going to:

`http://yourblog.com/?/wlmapi/2.0/return_format/resource_name`

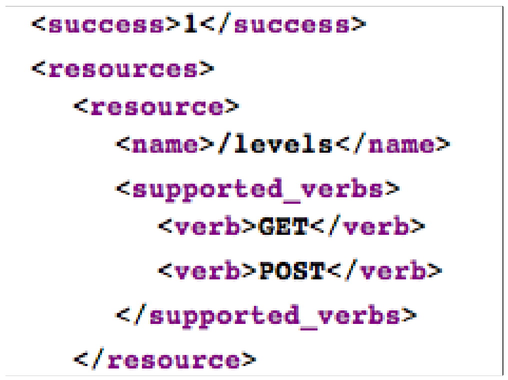
Where:

`yourblog.com`	URL where you installed WordPress that's running WishList Member
`return_format`	Can be xml, php or json
`resource_name`	The resource you wish to access

Give it a try by going to any of the following URLs in your web browser (be sure to replace yourblog.com with the correct URL):

Return Format	URL
XML	`http://yourblog.com/?/wlmapi/2.0/xml/resources`
PHP Serialized	`http://yourblog.com/?/wlmapi/2.0/php/resources`
JSON	`http://yourblog.com/?/wlmapi/2.0/json/resources`

If you chose XML, then you should get output similar to the following:



```
<success>1</success>
<resources>
   <resource>
      <name>/levels</name>
      <supported_verbs>
         <verb>GET</verb>
         <verb>POST</verb>
      </supported_verbs>
   </resource>
```

What we just did was made a GET request to the API using your browser and yes, it's that easy.

wlmapiclass.php: Making Things Easier

To make things easier for third-party developers, we developed a PHP class to handle most of the hard stuff in using the API. This class makes use of the cURL library to handle HTTP communication with the WishList Member API as well as all authentication requirements.

You can download the API Class for free at:

http://wishlistproducts.com/wp-content/uploads/2011/07/wlmapiclass.zip

To use the API class, you will need to put the following code at the top of your application.

```
<?php
      include('wlmapiclass.php');
      $api = new wlmapiclass('http://yourblog.com/', 'yourAPIKey');
      $api->return_format = 'php'; // <- value can also be xml or json
?>
```

Replace:

`http://yourblog.com/` with the full URL of your WordPress installation.

`yourAPIKey` with the WishList Member API key found under the settings tab

Yup, that easy. Initializing the class handles all authentication requirements between your application and the WishList Member API.

It's best to put those 3 lines of code in a separate config file and include it on top of your application.

Support for Other Programming Languages

WishList Member only provides a PHP version of **wlmapiclass** at the moment and will only support official versions of this class. You are however allowed and encouraged to translate the class to any language of your choice if you need to.

API Class Methods

The WishList Member API makes use of the Create, Retrieve, Update and Delete (CRUD) model in managing data. To complement this, the API Class has four (4) methods that you can use to communicate with the WishList Member API. Each method represents one of the four (4) HTTP verbs that the API supports, namely:

API Method	HTTP Verb	Action
$api->post	POST	Create
$api->get	GET	Retrieve
$api->put	PUT	Update
$api->delete	DELETE	Delete

With this four methods/verbs, it is possible to manipulate all data that is exposed by WishList Member through its API.

POST: Creating New Data

Where possible, WishList Member allows you to create new data through the API by making am HTTP POST request to a specific resource.

Syntax:

```
$response = $api->post($resource , $data);
```

Where:

$resource	- an API resource (i.e. /levels)
$data	- an associative array of data to pass to the API

Returns:

$response	- string: can be either serialized PHP data, JSON or XML

Example:

To create a new Membership Level using the API.

```
<?php
	$data = array('name' => 'Silver Level');
	$response = $api->post('/levels', $data);
?>
```

GET: Retrieving Data

The GET method is most probably going to be the most used method when using the API. It allows you to retrieve data specific to a resource.

Syntax:

```
$response = $api->get($resource [, $data]);
```

Where:

$resource	- an API resource (i.e. /levels)
$data	- (optional) an associative array of data to pass to the API

Returns:

$response	- string: can be either serialized PHP data, JSON or XML

Example:

To retrieve a list of all membership levels

```
<?php
      $response = $api->get('/levels');

      // we unserialize the response because we're using PHP as return format
      $response = unserialize($response);
      // dump the response to output
      print_r($response);
?>
```

To retrieve details of a particular membership level

```
<?php
      $level_id = 1234567890;
      $response = $api->get('/levels/'.$level_id);

      // we unserialize the response because we're using PHP as return format
      $response = unserialize($response);
      // dump the response to output
      print_r($response);
?>
```

PUT: Updating Data

The PUT method allows you to update the data in resources that support it.

Syntax:

```
$response = $api->put($resource , $data);
```

Where:

$resource	- an API resource (i.e. /levels)
$data	- an associative array of data to pass to the API

Returns:

$response	- string: can be either serialized PHP data, JSON or XML

Example:

To change the name of a membership level

```
<?php
	$level_id = 1234567890;
	$data = array('name' => 'New Level Name');
	$response = $api->put('/levels/'.$level_id, $data);
?>
```

DELETE: Deleting Data

The DELETE method will let you delete specific resources such as members, membership levels, etc. Note that this action cannot be reversed so use it with care.

Syntax:

```
$response = $api->delete($resource);
```

Where:

$resource	- an API resource (i.e. /levels)

Returns:

$response	- string: can be either serialized PHP data, JSON or XML

Example:

To delete a membership level

```
<?php
	$level_id = 1234567890;
	$response = $api->delete('/levels/'.$level_id);
?>
```

Checking for a Successful Request

An API request can either by successful or not. The best way to explain how to check for this is through code so let's go through it.

```
<?php
      $response = $api->get('/levels');
      $response = unserialize($response);

      if($response['success']==1){
            echo 'Request successful';
            // do stuff here
      } else {
            echo 'Request failed';
            echo '<br />';
            echo 'Error Code:' . $response['ERROR_CODE'];
            echo '<br />';
            echo 'Error Description:' . $response['ERROR'];
      }
?>
```

Checking for Supported Verbs

Each request to an API Resource except for the **`/resources`** and **`/auth`** resources will also return a list of supported verbs. This information is returned in the **`supported_verbs`** array. This is helpful if you want to make a quick check on what kind of actions you can do with a resource.

```
<?php
      $response = $api->get('/levels');
      $response = unserialize($response);

      print_r($response['supported_verbs']);
?>
```

TIP: You can also just call **`/resources`** and get a full list of available resources and their supported verbs.

API Resources

Membership Levels

This set of resources allows remote management of membership levels

Resource: /levels

Supported Verbs: GET, POST

GET

Retrieves an array list of all membership levels. Each entry contains the following:

Variables:

Variable Name	Type	Description
$id	integer	Membership Level ID
$name	string	Membership Level Name

POST

Creates a new membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$id	integer	**READ-ONLY** Membership Level ID	
$name	string	**REQUIRED** Membership Level Name	
$registration_url	string	Registration URL slug. Auto-generated if not specified on creation	auto-generated
$after_login_redirect	mixed	After login redirect page Possible values are ***"global"*** - default global settings; ***"homepage"*** - homepage; ***$page_id*** - specific page	global
$after_registration_redirect	mixed	After registration redirect page Possible values are ***"global"*** - default global settings; ***"homepage"*** - homepage; ***$page_id*** - specific page	global
$access_all_pages	boolean		FALSE
$access_all_categories	boolean		FALSE
$access_all_posts	boolean		FALSE
$access_all_comments	boolean		FALSE
$no_expiry	boolean		TRUE
$expiry	integer	Expiration schedule (ignored if $no_expiry is TRUE)	
$expiry_period	string	The period relating to $expiry Possible values are ***Days***; ***Weeks***; ***Months***; ***Years***	Days
$sequential_upgrade_to	integer	Membership Level ID to upgrade to. Leave blank to disable sequential upgrade for level	
$sequential_upgrade_after	integer	Sequential upgrade schedule in days	

$sequential_upgrade_method	string	Sequential upgrade method Possible values are ***ADD***; ***MOVE*** Note: Zero-day moves are not allowed	
$member_count	integer	**READ-ONLY** Total number of members in the level	
$require_captcha	boolean		FALSE
$require_email_confirmation	boolean		FALSE
$require_admin_approval	boolean		FALSE
$grant_continued_access	boolean		FALSE
$disable_existing_users_link	boolean		FALSE
$registration_date_reset	boolean		FALSE
$uncancel_on_registration	boolean		FALSE
$wordpress_role	string	WordPress role	subscriber
$level_order	integer		
$remove_from_levels[remove_from_level]	array	An array of membership level Ids to remove a user from when a user is added to this level.	

Resource: /levels/{$level_id}

Supported Verbs: GET, PUT, DELETE

GET

Retrieves full information about an individual membership level.

PUT

Updates the membership level specified by $level_id in the resource URL

Refer to **Resource: /levels → POST → Accepted Variables** for a list of editable variables.

DELETE

Deletes the membership level specified by $level_id in the resource URL

Level Members

This set of resources allows remote management of the members in each membership level

Resource: /levels/{$level_id}/members

Supported Verbs: GET, POST

GET

Retrieves an array list of all members in a membership. Each entry contains the following:

Variables:

Variable Name	Type	Description
$id	integer	Member ID
$user_login	string	User Login (username)
$user_email	String	User's email address

POST

Adds a member to a membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$Users	array	**REQUIRED** An array of Member Ids to add to the membership level	
$Cancelled	boolean	Cancellation status	FALSE
$CancelDate	integer	Unix timestamp of a scheduled cancellation date	
$Pending	boolean	Require Admin Approval	FALSE
$UnConfirmed	boolean	Email address confirmation status	FALSE
$Expired	boolean	**READ-ONLY**	
$ExpiryDate	integer	**READ-ONLY** Date when membership level will expire. Dynamically computed based on the $Timestamp and the membership level's expiry settings	
$Active	boolean	**READ-ONLY**	
$Status	string	**READ-ONLY** Possible values are ***"Active"***, ***"Unconfirmed"***, ***"For Approval"***, ***"Cancelled"***, ***"Expired"***	
$Timestamp	integer	Unix timestamp of when the member was added to the membership level	Current Date
$TxnID	string	The transaction ID	Auto-generated

Resource: /levels/{$level_id}/members/{$member_id}

Supported Verbs: GET, PUT, DELETE

GET

Retrieves membership level information for an individual member

PUT

Updates membership level information for an individual member

Refer to **Resource: /levels → POST → Accepted Variables** for a list of editable variables.

DELETE

Removes the member from the membership level

Level Content

This set of resources allows remote management of content; posts, pages, comments and categories; in each membership level

Resource: /levels/{$level_id}/posts

Supported Verbs: GET, POST

GET

Retrieves an array list of all posts in a membership level. Each entry contains the following:

Variables:

Variable Name	Type	Description
$ID	integer	Post ID
$name	string	Post Title

POST

Adds a post to a membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Post Ids to add to the membership level	

Resource: /levels/{$level_id}/posts/{$post_id}

Supported Verbs: DELETE

DELETE

Removes a post from a membership level

Resource: /levels/{$level_id}/pages

Supported Verbs: GET, POST

GET

Retrieves an array list of all pages in a membership level. Each entry contains the following:

Variables:

Variable Name	Type	Description
$ID	integer	Page ID
$name	string	Page Title

POST

Adds a page to a membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Page Ids to add to the membership level	

Resource: /levels/{$level_id}/pages/{$page_id}

Supported Verbs: DELETE

DELETE

Removes a page from a membership level

Resource: /levels/{$level_id}/comments

Supported Verbs: GET, POST

GET

Retrieves an array list of all posts with comments in a membership level. Each entry contains the following:

Variables:

Variable Name	Type	Description
$ID	integer	Post ID
$name	string	Post Title

POST

Adds a post's comments to a membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Post Ids to add to the membership level	

Resource: /levels/{$level_id}/comments/{$post_id}

Supported Verbs: DELETE

DELETE

Removes a post's comments from a membership level

Resource: /levels/{$level_id}/categories

Supported Verbs: GET, POST

GET

Retrieves an array list of all categories in a membership level. Each entry contains the following:

Variables:

Variable Name	Type	Description
$ID	integer	Category ID
$name	string	Category Name

POST

Adds a category to a membership level

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Category Ids to add to the membership level	

Resource: /levels/{$level_id}/categories/{$category_id}

Supported Verbs: DELETE

DELETE

Removes a category from a membership level

Content Protection

This set of resources allows remote management of the protection settings for posts, pages and categories.

Resource: /protected/posts

Supported Verbs: GET, POST

GET

Retrieves an array list of all posts that are protected

Variables:

Variable Name	Type	Description
$ID	integer	Post ID
$name	string	Post Title

POST

Protects a post

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Post Ids to Protect	

Resource: /protected/posts/{$post_id}

Supported Verbs: DELETE

DELETE

Unprotects a post

Resource: /protected/pages

Supported Verbs: GET, POST

GET

Retrieves an array list of all pages that are protected

Variables:

Variable Name	Type	Description
$ID	integer	Page ID
$name	string	Page Title

POST

Protects a page

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Page Ids to Protect	

Resource: /protected/pages/{$page_id}

Supported Verbs: DELETE

DELETE

Unprotects a page

Resource: /protected/categories

Supported Verbs: GET, POST

GET

Retrieves an array list of all categories that are protected

Variables:

Variable Name	Type	Description
$ID	integer	Category ID
$name	string	Category Name

POST

Protects a category

Accepted Variables:

Variable Name	Type	Description	Default Values
$ContentIds	array	**REQUIRED** An array of Category Ids to Protect	

Resource: /protected/categories/{$category_id}

Supported Verbs: DELETE

DELETE

Unprotects a category

Members

This set of resources allows remote management of users

Resource: /members

Supported Verbs: GET, POST

GET

Retrieves an array list of all members from the database. Each entry contains the following:

Variables:

Variable Name	Type	Description
$id	integer	Member ID
$user_login	string	User Login (username)
$user_email	String	User's email address

POST

Adds a new member to the database.

Accepted Variables:

Variable Name	Type	Description	Default Values
$user_login	string	**REQUIRED** User login (username)	
$user_email	string	**REQUIRED** User's email address	
$user_pass	string	Password. Auto-generated if not specified on user creation.	auto-generated
$company	string	Company name	
$address1	string	Address Line #1	
$address2	string	Address Line #2	
$city	string	City	
$state	string	State	
$zip	string	Zip Code	
$country	string	Country	
$wpm_login_limit	integer	Number of concurrent logins to allow per IP	
$wpm_registration_ip	string	IP address during user registration	
$custom_*{field}*	string	Custom member fields. Custom fields can be added by prepending a variable name with "custom_" (i.e. $custom_phone)	
$Sequential	boolean	Enables/Disables Sequential upgrade for user	
$Levels	array	Array of Membership Level Ids to add the user to	
$RemoveLevels	array	Array of Membership Level Ids to remove the user from.	

Note: Also accepts any other variable supported by the WordPress wp_update_user function.

Resource: /members/{$member_id}

Supported Verbs: GET, PUT, DELETE

GET

Retrieves membership details from the database

PUT

Updates a member's details

Refer to **Resource: /members → POST → Accepted Variables** for a list of editable variables.

DELETE

Deletes a member from the database

Old API Access

This allows access to the old WishList Member API methods

Resource: /api1/{$api1function}

Supported Verbs: GET

GET

Makes a call to the old API by passing function parameters in the Params[] array as query variables. The sequence of each element in the $Params array matches the sequence of parameters in the called method.

Example:

To call WLMAPI::GetUserLevels via for user ID 123, the request would be:

```
http://yourblog.com/?/wlmapi/2.0/api1/GetUserLevels?Params[]=123
```

Sample Code:

```
<?php
      $data=array(
            'Params'=>array(123)
      );
      $api->get('/api1/GetUserLevels',$data);
?>
```

Click here for a list of all old API methods.