WishList Member API 1.0 Documentation

WishList Member provides an application programming interface (API) that allows other applications to interface with a wide variety of its features. This API will allow you to write both internal extensions and external scripts that can access WishList Member data and functions and open up a range of possibilities for integrating with, extending, and programming around WishList Member. Below is the documentation for the WishList Member API.

Extensions

Extensions that communicate with WishList Member via the API can be written by 3rd party developers to enhance the core features already provided by WishList Member. This can be easily facilitated by creating a PHP script and dropping it in the WishList Member plugin's extensions folder. All scripts found in this folder will be executed by WishList Member upon execution of it's WordPress "init" hook.

The basic structure of a WishList Member Extension is like this:

```
<?php
// Extension Information
$WLMExtension = array (
'Name' => 'Extension Name',
'URL' => 'http://extension-website.com/',
'Version' => '1.23',
'Description' => 'Description of the extension',
'Author' => 'Extension Author',
'AuthorURL' => 'http://www.extension-author-website.com/',
'File' => __FILE__
);

if (!class_exists('MyWLMExtension')) {
class MyWLMExtension { // <- this is your class name
var $mode;
function MyWLMExtension () {
```

```
$this->mode = basename (__FILE__); // <- we save filename in $this->mode
}
function SettingsPage ($mode) {
if ($mode != $this->mode) return false; // <- not us, return
// it's us so we take action
// show your extensions' settings page here
}
}
}

if (!isset ($MyWLMExtension){
// setup our extension class
$MyWLMExtension = new MyWLMExtension ();
// we hook our settings page
add_action ('wishlistmember_extension_page', array(&$MyWLMExtension,'SettingsPage'));
}
?>
```

Remote Applications

The same functions available to WishList Member extensions can be made available to remote applications via a simple to use REST interface. This allows developers to create a lot of different applications that control a particular WishList Member powered website remotely.

The REST call is made like this:

http://yourblog.com/?WLMAPI=FxnName/Key/Parameter1/Parameter2a,Parameter2b,Parameter2c/Paramater3

FxnName – name of function to call (i.e. AddUserLevels)

Key – the MD5 hash key. This key is generated as follows:

```
<?php
 $Fxn = 'AddUserLevels';
 $Secret = 'TheSecretKey';
```

```
 $params = array ('Parameter1', 'Parameter2a,Parameter2b,Parameter2c', 'Parameter3');
 $Key = md5 ($Fxn . '__' . $Secret . '__' . implode ('|',$params));
?>
```

A successful REST call will return a serialized array containing two elements. The First element is always TRUE for successful calls and the Second element is the value that was returned by the function.

A failed REST call on the other hand returns a serialized array also containing two elements. The First is always FALSE and the Second element is a description of the Error that occurred.

The following code shows how to interpret a REST return value:

```
<?php
 list ($status, $return) = unserialize ($RESTReturn);
 if ($status) {
 // successful – do something useful here
 var_dump ($return);
 } else {
 // failed, show error message
 echo $return;
 }
?>
```

WLMAPI Class Methods

The WLMAPI Class Methods are the actual API functions that extensions and remote applications call. It is currently composed of a number of functions that allow developers to Get, Add, and Delete users, posts, pages, categories, and comments; as well as, functions for changing user and level settings, managing members, and accessing important user, level, and settings data.

WLMAPI::GetOption

GetOption(string $option)

Get various WLM option settings using this function. This functions requires a parameter $option

which is the option to be retrieved. Use this function to get the settings for:

1. register_email_body
2. register_email_subject
3. email_sender_name
4. email_sender_address

This function returns the value of the requested setting.

WLMAPI::GetLevels

GetLevels()

Get all registered membership levels. This function accepts no parameters. It returns a multi-dimensional array of all the membership levels created in WLM and the settings for each level. The first-level array key is the SKU for each respective membership level. The first-level array value is an array whose array key is the name of each setting and whose value is the value of each setting.

WLMAPI::AddUser

AddUser(string $username, string $email, string $password, [string $firstname=''], [string $lastname=''])

Add a new user to WishList Member. This functions requires 5 parameters:

1. string $username – the username for the new user
2. string $email – the email address for the new user
3. string $password – the password for the new user
4. string $firstname – the first name of the new user
5. string $lastname – the last name of the new user

This functions returns the new user's user ID on success and False on failure.

WLMAPI::EditUser

EditUser(integer $id, [string $email = ''], [string $password = ''], [string $firstname = ''], [string $lastname = ''], [string $displayname = ''], [string $nickname = ''])

Edit an existing member in WishList Member. This function requires 7 parameters:

1. int $id – the user id of the target user
2. string $email – the email address for the target user
3. string $password – the password for the target user
4. string $firstname – the first name of the target user
5. string $lastname – the last name of the target user
6. string $displayname – the display name of the target user
7. string $nickname – the nickname of the target user

This functions returns the target user's user ID on success and False on failure.

WLMAPI::DeleteUser

DeleteUser(integer $id, [integer $reassign = null])

Delete an existing user in WishList Member. This function accepts two parameters:

1. int $id – the user id of the user to be deleted
2. int $reassign – (optional) the user id to reassign all posts and links to

This function returns True on success and the description of the error that occurred on failure.

WLMAPI::GetUserLevels

GetUserLevels(int $user, [string $levels = 'all', [string $return = 'names', [bool $addpending = false, [bool $addsequential = false, [$cancelled = 0]]]]])

Get the membership levels for a user. This function accepts two parameters:

1. int $user – user id of the user to check
2. string $levels – (optional) 'all' to return all levels that a user is a member of OR a comma delimited string of level names or skus. Default is 'all'
3. string $return – (optional) either 'names' to return level names OR 'skus' to return level IDs. Default is 'names'
4. bool $addpending – (optional) TRUE to add pending status to array. Default is FALSE
5. bool $addsequential – (optional) TRUE to add sequential status to array. Default is FALSE
6. int $cancelled – (optional) 0 to not return cancelled levels. 1 to return names of cancelled levels with strikethrough. 2 to return names of cancelled levels without strikethrough. Default is 0

This functions returns an associative array on success and False on failure. Array keys are the Level IDs and the array values are the Level Names.

WLMAPI::AddUserLevels

AddUserLevels(int $user, array $levels, [string $txid = '', [bool $autoresponder = false]])

Add the user to the specified levels. This function accepts three parameters:

1. int $user – the user id of the target user
2. array $levels – an array of the level ids to add the target user to
3. string $txid – (optional) Transaction ID to for the added level. Default is ”
4. bool $autoresponder – (optional) set to TRUE if user is to be subscribed to the autoresponder for the specified levels. Default is FALSE

This functions returns TRUE on success and FALSE on failure.

WLMAPI::DeleteUserLevels

DeleteUserLevels(int $user, array $levels, [bool $autoresponder = true])

Remove the user from the specified levels. This function accepts three parameters:

1. int $user – the user id of the target user
2. array $levels – an array of the level ids to remove the target user from
3. bool $autoresponder – set to FALSE if user is to remain subscribed to the autoresponder for the specified levels. Default is TRUE

This functions returns TRUE on success and FALSE on failure.

WLMAPI::GetMembers

GetMembers()

Get an array of levels with a string of members, plus pending and non-sequential. This function accepts no parameters.

This function returns an array. The array keys are each level’s SKU plus pending and nonsequential. The array values for each level key are a string of the member IDs associated with that level. The array values for the pending and nonsequential keys are a string of member

IDs who are pending and nonsequential respectively.

WLMAPI::MergedMembers

MergedMembers(string $levels, bool $strippending)

Get a list of members in one or more levels. This function can be used with a single level in the $levels parameter to determine all the members for that level or it can be used with two or more levels in the $levels parameter to determine what members belong to all the designated levels. This function accepts two parameters:

1. string $levels – the levels to evaluate. Either “all” for all registered levels or a comma-delimited string of the levels to evaluate
2. bool $strippending – set to TRUE to strip pending members from the return. Default is FALSE

This function returns a comma-delimited string of member ids for all members in matching levels.

WLMAPI::GetMemberCount

GetMemberCount(string $level)

Get a count of all members in a level or levels. This function accepts a single parameter:

1. string $level – the level to evaluate. Can be “all”, “nonmembers”, “pending”, or a comma-delimited string of level SKUs or names to evaluate.

This function returns an integer count of the number of members in the designated level or levels.

WLMAPI::MakePending

MakePending(int $id)

Make a member or members pending. This function requires one paramater:

1. int $id – the ID or array of IDs of the members to put into pending status.

This function returns the count of the IDs that were made pending.

WLMAPI::MakeActive

MakeActive(int $id)

Make a member or members active. This function requires one paramater:

1. int $id – the ID or array of IDs of the members to put into active status.

This function returns the count of the IDs that were made active.

WLMAPI::MakeSequential

MakeSequential(int $id)

Make a member or members sequential. This function requires one paramater:

1. int $id – the ID or array of IDs of the members to put into the sequential upgrade process.

This function returns the count of the IDs that were made sequential.

WLMAPI::MakeNonSequential

MakeNonSequential(int $id)

Make a member or members non-sequential. This function requires one parameter:

1. int $id – the ID or array of IDs of the members to take out of the sequential upgrade process..

This function returns the count of the IDs that were made non-sequential.

WLMAPI::MoveLevel

Move a member or members to a new level. Can only move members to a new level if they belong to one level, because otherwise we don't know which membership level to remove. This function requires two parameters:

1. int $id – user ID or array of user IDs to move to the new level
2. string $lev – SKU or name of level to change member or members to

This function returns the count of IDs successfully moved to the new level.

WLMAPI::CancelLevel

Cancel a member or members from a level. This function requires two parameters:

1. int $id – user ID or array of user IDs to cancel from the level
2. string $lev – SKU or name of level to cancel member or members from

This function returns a count of the IDs successfully cancelled from the level.

WLMAPI::UnCancelLevel

Uncancel a member or members from a level. This function requires two parameters:

1. int $id – user ID or array of user IDs to uncancel from the level
2. string $lev – SKU or name of level to uncancel member or members from

This function returns a count of the IDs successfully uncancelled from the level.

WLMAPI::GetPostLevels

GetPostLevels(integer $id)

Retrieve the membership levels that have access to the target post. This function accepts one parameter:

1. int $id – the target post ID

This functions returns an array containing the levels that have access to the target post.

WLMAPI::AddPostLevels

AddPostLevels(int $id, array $levels)

Adds the target post to the specified levels. This function accepts two parameters:

1. int $id – the target post ID
2. array $levels – the level IDs to add the target post to

This function always returns TRUE.

WLMAPI::DeletePostLevels

DeletePostLevels(int $id, array $levels)

Removes the target post from the specified level. This function accepts two parameters:

1. int $id – the target post ID
2. array $levels – the level IDs to remove the target post from

This function always returns TRUE.

WLMAPI::GetPageLevels

GetPageLevels(integer $id)

Retrieve the membership levels that have access to the target page. This function accepts one parameter:

1. int $id – the target page ID

This functions returns an array containing the levels that have access to the target page.

WLMAPI::AddPageLevels

AddPageLevels(int $id, array $levels)

Adds the target page to the specified levels. This function accepts two parameters:

1. int $id – the target page ID
2. array $levels – the level IDs to add the target page to

This function always returns TRUE.

WLMAPI::DeletePageLevels

DeletePageLevels(int $id, array $levels)

Removes the target page from the specified level. This function accepts two parameters:

1. int $id – the target page ID
2. array $levels – the level IDs to remove the target page from

This function always returns TRUE.

WLMAPI::GetCategoryLevels

GetCategoryLevels(integer $id)

Retrieve the membership levels that have access to the target category. This function accepts one parameter:

1. int $id – the target category ID

This functions returns an array containing the levels that have access to the target category.

WLMAPI::AddCategoryLevels

AddCategoryLevels(int $id, array $levels)

Adds the target category to the specified levels. This function accepts two parameters:

1. int $id – the target category ID
2. array $levels – the level IDs to add the target category to

This function always returns TRUE.

WLMAPI::DeleteCategoryLevels

DeleteCategoryLevels(int $id, array $levels)

Removes the target category from the specified level. This function accepts two parameters:

1. int $id – the target category ID
2. array $levels – the level IDs to remove the target category from

This function always returns TRUE.

WLMAPI::GetCommentLevels

GetCommentLevels(integer $id)

Retrieve the membership levels that have access to the target post/page comments. This

function accepts one parameter:

1. int $id – the target post/page ID

This functions returns an array containing the levels that have access to the target post/page comments.

WLMAPI::AddCommentLevels

AddCommentLevels(int $id, array $levels)

Adds the target post/pages comments to the specified levels. This function accepts two parameters:

1. int $id – the target post/page ID
2. array $levels – the level IDs to add the target post/page comments to

This function always returns TRUE.

WLMAPI::DeleteCommentLevels

DeleteCommentLevels(int $id, array $levels)

Removes the target post/page comments from the specified level. This function accepts two parameters:

1. int $id – the target post/page comments ID
2. array $levels – the level IDs to remove the target post/page comments from

This function always returns TRUE.

WLMAPI::PrivateTags

PrivateTags(string $content)

Passes a string through the WishList Member Private Tags processor. Allows you to use private tags outside the post content section. This function accepts one parameter:

1. string $content – the content to be protected using private tags

This function will return the string if the current user has access to the designated membership

level.

WLMAPI::ShowWLMWidget

ShowWLMWidget(array $widgetargs)

Displays the WishList Member sidebar widget. This function accepts one parameter:

1. array $widgetargs – the arguments for customizing the widget

This function displays the sidebar login widget.

WishList Member Action and Filter Hooks

You can also modify the behavior of WishList Member to some extent by using hooks.