Remote API

This section should help a 3rd party developer understand how the remote API system works. If you have any questions that are not answered here, feel free to send an e-mail to support@geodesicsolutions.com and we would be happy to address any specific questions you might have.

Remote API Calls

You interact with the Remote API by making a remote API call. There are different types of calls that do different things, for instance anything related to use sessions, would start with "session.".

Below is a run down of all the current Geo API calls that are built into the main software, and a brief summary of what they are used for. For more information on using any of the remote API calls, look at the PHP source code of the applicable "sample file".

Remote API Call	Sample Client File [1]	Description
misc.echo	misc.echo_client.php	Returns whatever variables you send it, useful for testing to make sure you are able to communicate with the remote API
session.init	session.init_client.php	Create or edit a session, and return the session ID, that you would then use to set a cookie "classified_session", assuming you are on the same domain name. Allows sending username/[plaintext pass OR user token] to log a user in.
session.touch	session.touch_client.php	Update the "last active" time for a particular session.
session.destroy	session.destroy_client.php	Removes a particular session. In other words, log a user out based on the session ID.
session.get	session.get_client.php	Gets an array of session data based on session ID, IP, and user agent
user.register	user.register_client.php	Register a new user. Note that the normal checks are still done (with the exception of the security image), and that registration "fields to use" still apply here, if you attempt to register and break the "fields to use" rules, an error will be returned.
user.edit	user.edit_client.php	Edits a user's information, such as their e-mail, first/last name, etc. Note that the registration "fields to use" still apply here, if you attempt to edit something and break the "fields to use" rules, an error will be returned.
user.exists	user.exists_client.php	Check to see if a username or e-mail exists or not.
user.getToken	user.getToken_client.php	Gets a user's remote API token that can then be used in other remote API calls, such as user.edit.

Remote API Call	Sample Client File [1]	Description
user.get	user.get_client.php	Gets a user's registration data, looked up by username OR e-mail address.
user.changeStatus	user.changeStatus_client.php	Uses a user's remote API token to change the status (login enabled/disabled) of that user
user.listings.active	user.listings.active_client.php	Gets a list of active listings for specific seller. API call added in **version 6.0.4**
user.listings.expired	user.listings.expired_client.php	Gets a list of expired listings for specific seller. API call added in **version 6.0.4**
user.listings.favorite	user.listings.favorite_client.php	Gets a list of favorite listings for specific user. API call added in **version 6.0.4**
system.getSetting	system.getSetting_client.php	Gets the value of a specified internal setting
users.count	users.count_client.php	Returns the number of registered users on the site
users.list	users.list_client.php	Returns a list of registered users on the site
category.list	category.list_client.php	Returns a list of categories on the site
groups.list	groups.list_client.php	Returns a list of user groups on the site
listing.search	listing.search_client.php	Searches the database of listings for a given keyword, optionally in a given category
listing.get	listing.get_client.php	Gets the details and images for a given listing

Using a Sample Client File

View the source on any of the sample client files. Each one will have in-line comments on how to use it, along with variables that you will have to set to use it.

Each sample file is a **fully working standalone script** that makes use of the applicable remote API call, but you must follow the instructions to set it up. For your reference, here are the "common" set of instructions that apply to most of the stand-alone sample scripts.

Code snippet from one of the files in classes/api/_samples/:

```
/*
Note: This is intended for people that are familiar with editing PHP
    files.

Instructions for using this as a stand-alone API client:
1.  Edit this file:  Look for the line that starts with "$xmlrpc_location".
    Change it to:
    $xmlrpc_location = "XMLRPC.class.php";
2.  Set the rest of the "Required Settings" as needed (like $website,
$api_key, etc)
    Each setting has its own instructions right above it.

    There may be optional settings as well, those can be set by
    un-commenting them and set them as instructed.
3.  Upload the modified file to a location that you can access from the web.
    It does not have to be on the same website as the Geo software.
4.  Upload the file "XMLRPC.class.php" to the same location that you
```

```
uploaded
    this file to.  The file is located in the Geo software at:
    classes/rpc/XMLRPC.class.php
5.  In a web browser, visit the file you uploaded in step 3.  You should see
    the results of the API call.
*/
```

Sample Setup

The initial steps would be to take one of the files within the remote samples directory found here in the file set:

```
/classes/api/_samples/
```

and move it to where you would like that api call to be on your site. Lets take the following file:

```
category.list_client.php
```

You can move to a specific location on your site where you want all your api calls to sit. You could rename that file to anything you like. Let say I renamed the file and placed it here on my site:

```
http://example.com/remote_api_calls/get_categories.php
```

Once there configure everything within the top of that file. Lets assume the Geo software is installed in the following location and that I have not moved the XMLRPC.class.php file. The configurations in that file would be:

```
//location of the XMLRPC.class.php file.  If calling the script from the
_samples folder, leave this setting at the default.
$xmlrpc_location = '../classes/rpc/XMLRPC.class.php';

//url of geo_api.php file, something like "https://mysite.com/geo_api.php"
//Note: recommended to use https for a secure connection, but if your server
does not have SSL, you can use http instead.
$website = "https://example.com/geo_api.php";

//Site's API key.  You can find the site's API key on the home page in the
admin.
$api_key = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';
```

You would pull the api key for the get_categories.php call (copied from "category.list_client.php") from the following configuration in the admin tool:

```
SITE SETUP > REMOTE API SECURITY KEYS > core.category.list
```

And you would paste that key directly into that file where the "x's" are above. That should be all the configurations in that file for that call. If all calls are in the same location the paths should be the same. Only the api key should change for each as each remote is different. The core.category.list call happens to require "parent" to be passed into it. That would be the parent category id of the

subcategories to get. So given the above urls the call to get the main categories would be:

```
http://example.com/remote_api_calls/get_categories.php?parent=0
```

Each call requires it's own data and supplies it's own data in the response. You'll need to get to know each to create a call for each.

Calls Created by Addons

Note that you can create a new API call, inside of an addon. As long as that addon is installed and enabled, you will be able to use the API call.

Before creating a new remote API call yourself, if it makes sense for the API call to be one of the built-in calls (for instance, an API call to get the listing data for a particular listing), but is just not yet created, contact us as support@geodesicsolutions.com to request it. Every single remote API call that is currently in existance, has been created because it is "in demand" (just like most of the current "addon hooks").

To create your own remote API call in your custom addon, create a sub-directory named "api" inside of the addon's directory. Then you would create a PHP file for any API call you want to make. Any files that are prepended with an underscore "_" will be ignored by the system. We recommend starting from the **misc.echo** remote API call and building on top of that. You can also see the Example Addon for more information about creating a new remote API call.

See Also

- [Remote API Security Keys](#) - Page that shows the Remote API security keys for every single API call, something you will need when making an API call. See any of the sample remote API clients in **classes/api/_samples/** for more info.

[1)](#)
each file found in directory **classes/api/_samples/**

From:
https://geodesicsolutions.org/wiki/ - **Geodesic Solutions Community Wiki**

Permanent link:
https://geodesicsolutions.org/wiki/developers/remote_api/start

Last update: **2015/07/10 17:47**