

Installation and Upgrades

Geo Mashup supports [standard WordPress methods](#) for installation and upgrades.

Configuration

After installing, activate the plugin (from the plugins list if you're not prompted). You should then go to Settings / Geo Mashup to configure it.

Basic Usage

Geo Mashup lets you save location information for various "objects" such as posts (including custom types), pages, users, and comments. You can then put these objects on maps in posts, pages, or just about anywhere else on your site.

Map Providers

Geo Mashup uses the [mapstraction library](#) to offer a choice of map providers. Visit the Geo Mashup settings to choose one in the Overall tab. Your choice has some consequences:

- Google Maps v3 now supports clustering. An [API key](#) is optional for this choice.
- OpenLayers no marker clustering, no API key required.
- Leaflet no marker clustering, no API key required.
- Google Maps v2 support has expired

Save locations for things like posts

By default Geo Mashup won't show a map unless there is something like a located post to put on it. It's most common to save locations for posts and pages, but Geo Mashup

can also save locations for custom post types, users, and comments. You can choose which of these you want to collect location for in the Overall settings - the default is posts and pages.

When adding or editing a post or page, a Location area has been added near the bottom of the editor page:

Location

Find a new location:

or select from

Saved Locations

↑

↓

↶

↷

+

-

Map

Satellite

Hybrid

Map data ©2011 Geocentre Consulting, MapLink, Tele Atlas - [Terms of Use](#)


Address	Saved Name	Geo Date
		Jul 9, 2011
		@ 15 : 19

[help](#)

You can drag the location area closer nearer to the top of the page if you like. Click in the Find Location textbox and type a place name (like Chartre in this example) to do a search:

Location

Find a new location: or select from



Address	Saved Name	Geo Date
Chartre, 42130 Saint-Étienne-le-Molard, France 45.7507004, 4.0893847	<input type="text"/>	<input type="text" value="Jul 9, 2011"/> <input type="text" value="@ 15 : 19"/> <input type="button" value="Save"/>

[help](#)

Notice the help link next at the bottom. Clicking this will display the instructions for other ways to search for a location. The location that will be saved is marked with a green pin. If there's already a location saved, you can change it or delete it.

Add a Map

You can put a map in any page or post by typing the shortcode `[geo_mashup_map]` into the editor. Look at the [Tag Reference](#) for more options.

Single Map

If the post or page has a location, the map displays a Single Map showing only that location.

Global Map

If the post or page has no location, a Global Map of all located posts and pages is displayed.

Contextual Map

You can put a map outside posts and pages by using the template tag `<?php echo GeoMashup::map(); ?>` to make a Contextual Map that shows the located items listed nearby. (Technically contextual maps include located items in the current global query results - usually that's the main list of posts, or the page being viewed). The [Tag Reference](#) has details for template tags.

Getting the wrong kind of map?

There are situations when Geo Mashup won't output the kind of map you want. You can specify one of the types with the `map_content` parameter, like `[geo_mashup_map map_content="single"]` or `<?php echo GeoMashup::map("map_content=global"); ?>`. See [Tag-Reference#query-variables](#) for details.

What's Next

Now your imagination might go wild with the possibilities. Skim the [Feature Usage](#) page to see what's available. When you're ready to add your map shortcodes and tags, look at the [Tag Reference](#) to see how far beyond a simple `[geo_mashup_map]` or `<?php echo GeoMashup::map(); ?>` you can go.

If you need help, you can offer to pay an amount of your choice for expert help at [WP Questions](#), or ask other users in the free [Google Group](#).

Please [submit an issue](#) if you run into problems, or would like to request a feature.

These are some of the common things you can do with Geo Mashup.

- [Add Widgets](#)
- [Cluster Markers](#)
- [Add Taxonomies](#)
- [Add a Global Map Page](#)

- [Use Auto Zoom](#)
- [Category Lines](#)
- [Use KML for richer maps](#)
- [Add a "show on map" link](#)
- [Customize the info window and other content](#)
- [More](#)

Widgets

Shortcodes will work in a text widget, so you can use `[geo_mashup_map]` and [Other Tags](#) within a widget.

Geo Mashup Search Widget

Geo Mashup can provide a geo search widget to find posts within a specified distance of a search location. A "find me" button allows a search near the user's current location.

1. Under Settings / Geo Mashup / Overall check the "Enable Geo Search" setting and update options.
2. Create a page where search results will be displayed.
3. Under Appearance / Widgets place the Geo Mashup Search widget where you would like it to appear.
4. Fill in the widget settings, making sure to select the results page you created in step 2, and save.

Clustering

Marker clustering is currently supported with Google maps only.

For global maps with many markers, it can be more meaningful to see how many markers are in an area than masses of overlapping markers. Clustering does this for

you. Because the problem is more severe at lower zoom levels, it's enabled by zoom level in the "Cluster Markers Until Zoom Level" setting for global maps.

Clusters are represented by a partially transparent red circle icon. The minimum cluster size is 4 markers, and clicking a cluster will zoom to that area in the map. [Custom javascript](#) code can modify these [MarkerClustererPlus options](#) using the 'markerClustererOptions' action.

Taxonomies

There is an Overall setting that allows you to choose which taxonomies are included on maps. The default is post category only.

Global Mashup Page

You can have a single page with a global map that will be the target of links generated by Geo Mashup.

Create a page, include the `[geo_mashup_map]` shortcode in it, and select the page in the Settings / Geo Mashup / Overall / Global Mashup Page dropdown list. The "Show on Map Link" tag and "Add Category Links" options will now create links to this page, which will load a global map appropriate for the link.

Auto Zoom

Zoom levels can be set to 'auto', in both settings and tags. The zoom level is calculated to show all the content on the map. A consequence of this is that the content must first be added to the map before the final zoom level is set, and this operation may be visible when the map loads.

Category Lines

At the bottom of Settings / Geo Mashup / Global Maps is a table of your categories. To enable connecting lines between posts of a category, select a color and enter a maximum zoom level for the category. Use a zoom level of 20 to always draw the line, or leave blank to never draw it.

Posts are connected in Geo Date order. By default the geo date is set to the post creation date, but you can change it in the post editor under Location / Geo Date.

KML Attachments

Instead of searching for a location in the location editor, you can upload a KML [file attachment](#) with a post or page using the "Add Media" button above the editor. Once uploaded, you can insert a link in your post or just close the upload window. The geographic center of all features in the KML file will be used to determine a default location for the post or page. Review this location in the location editor, adjust if needed, and save it.

Note: Google maps can only load KML files that are accessible to its servers, so a KML attachment on a private domain (e.g. localhost) or site will not work with Google.

Displaying a KML file

Once a page or post has a KML file attached, that file will be loaded in a single map.

You can also display KML files associated with the selected marker on a global map by checking "Show Geo Attachments" in Settings / Geo Mashup / Global Maps, or setting the `marker_select_attachments` parameter of the map tag to "true".

Creating a KML file

You can create a map using [Google My Maps](#) and save it as KML using [these instructions](#).

Show on map links

You can add a link from any located item to your designated global map page, with the global map centered on the clicked item. Use the `[show_on_map_link]` shortcode tag for a single item, or the template tag `<?php echo GeoMashup::show_on_map_link(); ?>` to add links to all posts or pages. See the [Tag Reference](#) for details.

Templating

The content in the info window is generated by a template that works just like any other [WordPress template](#). Geo Mashup comes with a default template found in `default-templates/info-window.php` that generates content similar to prior versions. Don't edit this, but you can copy it to `geo-mashup-info-window.php` in your theme directory, and your custom template will be preserved in future upgrades. There are templates for other types of info window content that can also be customized by copying to your theme folder:

- Copy `map-frame.php` to `geo-mashup-map-frame.php`
- Copy `comment.php` to `geo-mashup-comment.php`
- Copy `full-post.php` to `geo-mashup-full-post.php`
- Copy `info-window-max.php` to `geo-mashup-info-window-max.php`
- Copy `user.php` to `geo-mashup-user.php`
- Copy `nearby-list.php` to `geo-mashup-nearby-list.php`
- Copy `search-form.php` to `geo-mashup-search-form.php`
- Copy `search-results.php` to `geo-mashup-search-results.php`

You can also change the styling of the info window and other map components, copy `css/map-style-default.css` to `map-style.css` in your theme folder to get started.

WordPress Background: [Templates](#) and [Template Tags](#)

More Feature Resources

- Most settings on the Geo Mashup settings page have a short description of the feature they enable. Those aren't repeated here, so look through them to be aware of top-level features.
- Look through the list of tags in the [Tag Reference](#) to see the growing variety of Geo Mashup capabilities.
- Code snippets [tagged wordpress-geo-mashup in Snipplr](#)
- [The Google Group](#) has many more snippets and wisdom compiled over the ages.
- You can see [a fairly comprehensive list of completed enhancements](#), with lots of related information.

APIs For More Customization

Most map customization is done via the [javascript API](#), while the [PHP API](#) applies mostly to map data.

About Tags

- [Tag formats](#)
 - [Shortcode](#)
 - [Template](#)

Available Tags

- [Category Name](#)
- [Full Post](#)
- [List Located Posts](#)
- [List Located Posts By Area](#)
- [Location Info](#)
- [Map](#)

- [Nearby List](#)
- [Post Coordinates](#)
- [Save Location](#)
- [Show on Map Link](#)
- [Tabbed Term Index](#)
- [Term Legend](#)
- [Visible Posts List](#)

Tag Formats

Tags in post and page content use standard WordPress [Shortcode format](#). Template tags have also been updated to use standard [template tag parameter format](#) in WordPress templates. These formats are not complicated - the following examples should be all you need to understand them.

Shortcode Parameters

Here's an example of a tag you might put into a post or page:

```
[geo_mashup_map height="200" width="400" zoom="2" add_overview_control="false" add_map_type_control="false"]
```

The tag is `geo_mashup_map`, after which any number of parameters are specified. Any parameters that are not specified will use a default value from the Geo Mashup Options. If the post or page has a location saved, single default settings are used, otherwise the global defaults are used.

Template Tag Parameters

To get a similar map in a template, use template tag syntax:

```
<?php echo
```

```
GeoMashup::map('height=200&width=400&zoom=2&add_overview_control=false&add_map_type_control=false');?>
```

Note the 'echo' before GeoMashup. This is PHP for "display it right here", and as such is commonly used with template tags.

Tags

Category Name

Inserts the name of the currently displayed map category, if any.

Shortcode Tag: [geo_mashup_category_name]

Template Tag: <?php echo GeoMashup::category_name(); ?>

Full Post

For use with a global map created with the [Map](#) tag.

Displays full post content for the currently selected marker of a global map.

Shortcode Tag: [geo_mashup_full_post]

Template Tag: <?php echo GeoMashup::full_post(); ?>

Parameters:

- for_map - the name of the map this tag should work with. Required if the map has a name, otherwise uses the first map on the page.

List Located Posts

Inserts a list of all posts with location information.

Shortcode Tag: `[geo_mashup_list_located_posts]`

Template Tag: `<?php echo GeoMashup::list_located_posts(); ?>`

Parameters: You can use [query variables](#) with this tag.

List Located Posts By Area

Generates a list of located posts broken down by country and administrative area (state/province). Country heading is only included if there is more than one country. Names should be in the blog language if available.

Shortcode Tag: `[geo_mashup_list_located_posts_by_area]`

Template Tag: `<?php echo GeoMashup::list_located_posts_by_area() ?>`

Parameters:

- `include_address` - true (omit for false). Includes post address when available.

You can also use [query variables](#) with this tag.

Location Info

Display information about the location saved for this item (post, page, comment, etc). Blank if the requested information isn't present for a location.

Shortcode Tag: `[geo_mashup_location_info]`

Template Tag: `<?php echo GeoMashup::location_info(); ?>`

Accepted Parameters:

- **fields** - Possible values: address, geo_date, lat, lng, locality_name, admin_code, country_code, postal_code, saved_name. The fields admin_name and country_name will work also, but be aware that this will trigger a GeoNames lookup the first name. Results are cached for subsequent queries. Comma delimited list of fields to print. Default is address.
- **separator** - Text to print between field values. Default is a comma.
- **format** - If supplied, used as a format template for output, similar to [PHP's sprintf\(\) format](#).
- **object_name** - Possible values: post, user, comment. If supplied, used with object_id to display information about an object other than the "current" contextual object.
- **object_id** - Integer ID. If supplied, used with object_name to display information about an object other than the "current" contextual object.

Map

Inserts a map.

When the tag is in a located post or page, Single Maps settings are used by default. In a post or page without a location, Global Maps settings are used. As a template tag, this is the behavior inside [The Loop](#). In all other locations, Contextual Maps settings are used. You can always change the default map content with the map_content parameter, such as `[geo_mashup_map map_content="global"]`.

Shortcode Tag: `[geo_mashup_map]`

Template Tag: `<?php echo GeoMashup::map() ?>`

This tag has a lot of accepted parameters, so we'll categorize them.

Query Variables

These affect the objects that are selected to be shown on a global map. A bounding box query on a post map, for example, will select only posts inside the bounding box to appear on the map.

Some fields, like `locality_name` and `postal_code`, will only work if reverse geocoding is enabled and the target objects have been successfully geocoded.

- `admin_code` - Code for an administrative area. In the United States this is the two letter state code.
- `country_code` - Two character ISO country code.
- `exclude_object_ids` - Comma separated list of IDs objects to exclude identified in combination with `object_name`.
- `limit` - A maximum number of items to map.
- `locality_name` - A locality name, usually city or town.
- `map_cat` - the ID of the category to display. Accepts a comma separated list of IDs and IDs preceded by a minus sign are excluded, like the [WordPress query_posts cat parameter](#).
- `map_content` - global, single, contextual, or a [WP_Query object](#). Overrides the default content of a map. The `WP_Query` object is only an option for the template tag, and will include only posts returned by the query on the map.
- `map_offset` - number of posts to displace or skip over. Default is 0.
- `map_post_type` - Possible values: post, page, or custom post types. A comma separated list of post types to include on the map. Default is "any".
- `minlat` - Smallest decimal latitude for a bounding box query.
- `maxlat` - Largest decimal latitude for a bounding box query.
- `minlon` - Smallest decimal longitude for a bounding box query.
- `maxlon` - Largest decimal longitude for a bounding box query.
- `near_lat` - Used with `near_lng` and `radius_mi` to include objects near a point.
- `near_lng` - Used with `near_lat` and `radius_mi` to include objects near a point.

- **object_name** - Possible values: post, user, comment. The type of objects to include on the map. Default is post, which includes pages as a type of post.
- **object_id** - Integer ID of a single object to include identified in combination with **object_name**.
- **object_ids** - Comma separated list of IDs objects to include identified in combination with **object_name**.
- **postal_code** - A postal code - ZIP in the US.
- **radius_km** - Same as **radius_mi**, but takes a value in kilometers instead of miles.
- **radius_mi** - Used with **near_lat** and **near_lng** to include objects near a point.
- **saved_name** - The 'Save As' name used for a location.
- **show_future** - true, false, or only. Includes future-dated posts. Default is false.
- **sort** - Possible values: label, object_id, geo_date. Order can affect the category legend or listing tags. Default depends on the **object_name** parameter.
- **tax_query** - a taxonomy query array in the format used by [WP_Query](#). Currently only usable with template tags (not shortcodes).

Map Options

These affect the appearance or behavior of the map.

- **add_map_control** - true or false. Adds zoom/pan controls.
- **add_google_bar** - true or false. Replaces the Google logo with Google's small map search interface. Default is false.
- **add_map_type_control** - Possible values: G_NORMAL_MAP, G_SATELLITE_MAP, G_SATELLITE_3D_MAP, G_HYBRID_MAP, G_PHYSICAL_MAP. Comma separated list of map types to include in the map type control. Not used by default (blank).
- **add_overview_control** - true or false. Adds the overview control.
- **auto_info_open** - true or false. Opens the info window of the most recent or link source post.
- **auto_zoom_max** - auto zoom only up to the specified zoom level. Used with **zoom="auto"**.

- background_color - A [Hex triplet RGB Color](#) to use for the background before map tiles load. Default is gray, C0C0C0.
- center_lat - decimal latitude. With center_lng, an initial center location for the map.
- center_lng - decimal longitude. With center_lat, an initial center location for the map.
- click_to_load - true or false. Activates the click-to-load feature described above. Default is false.
- click_to_load_text - the text displayed in the click-to-load pane.
- cluster_max_zoom - Integer zoom level. The highest level to cluster markers. Not used by default (blank).
- enable_scroll_wheel_zoom - true or false. Enables the use of a mouse scroll wheel to zoom the map.
- height - the height of the map in pixels
- load_empty_map - true or false. Loads a map even if the query for markers return no results. Default is false.
- load_kml - url of a KML file. Displays the KML file on the map.
- map_control - !GSmallZoomControl, !GSmallMapControl, !GLargeMapControl, !GSmallZoomControl3d, !GLargeMapControl3D
- map_type - G_NORMAL_MAP, G_SATELLITE_MAP, G_HYBRID_MAP, G_PHYSICAL_MAP
- marker_min_zoom - hide markers until zoom level, 0 to 20
- marker_select_info_window - true or false. Enables the opening of the info window when a marker is selected. Default is true.
- marker_select_highlight - true or false. Enables highlighting of a marker when it is selected. Default is false.
- marker_select_center - true or false. Enables centering of a marker when it is selected. Default is false.

- `marker_select_attachments` - true or false. Enables loading of related KML attachments for a marker when it is selected. Default is false.
- `open_object_id` - The ID of an object to select when the map loads. Replaces `open_post_id` in earlier versions.
- `remove_geo_mashup_logo` - true or false. Removes the Geo Mashup logo from maps (this will be the default in future versions). Default is false.
- `static` - true or false. Attempts to use the Google Static Maps API to create a fast-loading static image map. Currently limited to the normal map type with one marker color and style.
- `width` - the width of the map. Use a plain number like 400 for pixels, or include a percentage sign like 85%.
- `zoom` - auto or valid integer zoom value. Auto will zoom to include all loaded map content. Default is auto.

Control Parameters

These affect the content and behavior of controls that interact with map content, like the category legend.

- `name` - name for this map. The `for_map` parameters of other tags is then used to tie them in with correct map in situations where multiple maps are possible.

Nearby List

List results of a radius search, centered on the current post by default. The default template is `geo-mashup/default-templates/nearby-list.php` - see comments there for instructions.

Shortcode Tag: `[geo_mashup_nearby_list]`

Template Tag: `<?php echo GeoMashup::nearby_list(); ?>`

Parameters:

- `exclude_object_ids` - Comma separated list of IDs objects to exclude identified in combination with `object_name`.
- `limit` - A maximum number of items to map.
- `location_text` - place name or address of search center. Not used by default.
- `map_cat` - the ID of the category to display. Accepts a comma separated list of IDs and IDs preceded by a minus sign are excluded, like the [WordPress query_posts cat parameter](#).
- `near_lat` - latitude of search center. Not used by default.
- `near_lng` - longitude of search center. Not used by default.
- `object_id` - the ID of a located object as search center. This object will be excluded from results. Default is the current loop object.
- `units` - mi or km. Whether search using miles or kilometers. Default is km.
- `radius` - number of units of search radius. Default is 50.
- `object_name` - post or user. Default is post.
- `sort` - Possible values: label, object_id, geo_date. Order can affect the category legend or listing tags. Default depends on the `object_name` parameter.
- `template` - base name of template to use for output. Default is 'nearby-list'.

Post Coordinates

This is currently only available as a template tag which you can use to display the coordinates stored for a page or post. Have a look at [Issue 134](#) for good examples of usage.

Template Tag: `<?php $coordinates_array = GeoMashup::post_coordinates() ?>`

Accepted Parameters:

- `places` - number of decimal places to include in coordinates. Default is 10.

Returns:

- An array containing 'lat' and 'lng' entries, so in the example `$coordinates_array['lat']` would contain latitude.

Save Location

This can be used to set the location for a post when the post editor location search interface is not available.

When a post or page is saved, the location is read from a special shortcode in the content. The location is saved, and the special shortcode is removed.

For now at least, decimals must be a period, not a comma.

Shortcode Tag: `[geo_mashup_save_location]`

Parameters:

- address - a location description to geocode (address, partial address, place name, etc). Required if used instead of the lat and lng parameters.
- geo_date - the date to associate with the location. A [variety of formats](#) will work. Defaults to current date.
- lat - decimal latitude. Required.
- lng - decimal longitude. Required.
- saved name - if you want the location added to the saved location menu, the name.

Show on Map Link

Inserts a link for the current post or page to overall blog map.

Shortcode Tag: `[geo_mashup_show_on_map_link]`

Template Tag: `<?php echo GeoMashup::show_on_map_link() ?>`

Accepted Parameters:

- `text` - the text of the link. Default is 'Show on map'.
- `show_icon` - true or false, whether to display the geotag icon before the link.
Default is true.
- `zoom` - the zoom level to start at, 0 to 20.

Tabbed Category Index

Deprecated, use Tabbed Term Index tag.

Tabbed Term Index

For use with a global map created with the [Map](#) tag.

Generates output that lists located posts by category in markup that can be styled as tabs. Subcategories are included in their parent category tab under their own heading. Click on post titles opens the corresponding marker's info window on the map. The map can show all markers, or just those for the active tab.

Additional CSS is required in the theme stylesheet to get the tab appearance, otherwise the markup will just look like a series of lists. An example is included in `tab-style-sample.css`.

Shortcode Tag: `[geo_mashup_tabbed_category_index]`

Template Tag: `<?php echo GeoMashup::tabbed_category_index() ?>`

Parameters:

- `for_map` - the name of the map this list should work with. Required if the map has a name, otherwise uses the first map on the page.

- taxonomy - a taxonomy identifier such as 'category'.
- show_inactive_tab_markers - true to show all markers, regardless of which tab is active. Default is false/omitted.
- start_tab_term - the ID of the tab term to select initially. Default is the first term loaded.
- tab_index_group_size - when set starts a new list element whenever this many terms are output. Default is off - a single group.
- disable_tab_auto_select - true to start with no tab selected. Default is false/omitted.

Term Legend

For use with a global map created with the [Map](#) tag.

Inserts a table or list of colored map pins and the taxonomy terms they go with (category by default). Checkboxes will show or hide category markers on the map.

Shortcode Tag: [geo_mashup_term_legend]

Template Tag: <?php echo GeoMashup::term_legend() ?>

Accepted Parameters:

- for_map - the name of the map this category legend should work with. Required if the map has a name, otherwise uses the first map on the page.
- taxonomy - a taxonomy identifier, limits the legend to only this taxonomy. Default is a separate legend section for each taxonomy included on the map.
- noninteractive - true to create a legend with show/hide checkboxes. Default is false/omitted.
- check_all - true or false, whether to add a check/uncheck all checkbox for each taxonomy. Default is true.

- `default_off` - true to load with all checkboxes unchecked so all markers are hidden on the map. Default is false/omitted.
- `format` - table, dl, or ul. Determines the HTML tags used for output: table, definition list, or unordered list. Default is table.
- `titles` - true to add a taxonomy title for each taxonomy section.

Term Order

Order will now be alphabetical by default.

You can use one of the many plugins available to change the term order. Since plugins use varying additional fields for ordering terms, I've made a simple extension method for changing the sort field. Just add a define to your `wp-config.php` or theme `functions.php`:

```
define( 'GEO_MASHUP_TERM_ORDER_FIELD', 'menu_order' );
```

Other plugins might use `'term_order'`, `'custom_order'`, etc.

Visible Posts List

Generates a list of post titles that are currently visible on the map. Clicking the title opens the info window for that post.

Shortcode Tag: `[geo_mashup_visible_posts_list]`

Template Tag: `<?php echo GeoMashup::visible_posts_list() ?>`

Parameters:

- `for_map` - the name of the map this list should work with. Required if the map has a name, otherwise uses the first map on the page.

As of version 1.6, WordPress 3.5 or higher is required. Geo Mashup is intended to support [the same PHP and MySQL versions as WordPress](#), but not all configurations are tested. Please [submit an issue](#) if you believe there's a problem with your version.

JavaScript and map providers

Until version 1.4, Geo Mashup used only v2 of the Google Maps API for maps, and custom javascript would be written only for that API. Version 1.4 still supports that API, but uses the Mapstraction library to enable more map providers, and now custom javascript can be written for Mapstraction or even tailored to specific map providers.

So if you have Geo Mashup javascript customizations for Google v2, they may not work with a new map provider. Calls to the [Google Maps API v2](#) need to be replaced with calls to the [mapstraction API](#) or the API of the provider you choose. The key changes are:

- `properties.map_api` identifies the map API in use, currently 'google', 'googlev3', 'leaflet', or 'openlayers'.
- `map` the map object is an instance of `mxn.Mapstraction` for 'googlev3' and 'openlayers'. Other objects like markers are instances of the corresponding mapstraction object.

To help keep custom javascript small and specific to a particular provider API, Geo Mashup loads the custom javascript file according to this file name scheme:

`custom-<provider>.js`

Where `<provider>` is currently google, googlev3, or openlayers. If that doesn't exist, but the provider is not google, this file is used for general Mapstraction customizations:

custom-mxn.js

And if that is not present or appropriate, the original is loaded:

custom.js

Back end editor customization

You may also add custom javascript for the back end location editor in the file:

location-editor.js

Usage

The API is based on an event interface that invokes callback functions for named actions. As an example we'll use the 'loadedMap' action to add a Flickr layer of squirrel pictures to the map.

If you're targeting the old Google v2 map provider, this code could go in a file named custom-google.js:

```
// An Example Google Maps API v2 customization
```

```
GeoMashup.addAction( 'loadedMap', function ( properties, map ) {  
    var kml;
```

```
    // Load some KML only into global maps - for instance pictures of squirrels
```

```
    if (properties.map_content == 'global') {
```

```
        // Make the Google Maps API v2 calls to add a Flickr KML layer
```

```
        kml = new
```

```
google.maps.GeoXml("http://api.flickr.com/services/feeds/geo/?g=52241987644@N01&lang=en-us&format=kml");
```



```
    map.addOverlay(kml);  
  
}  
  
} );
```

Any other map provider can be targeted in a custom-mxn.js file, but it will only work if the provider supports the features used:

```
// An Example Mapstraction customization
```

```
GeoMashup.addAction( 'loadedMap', function ( properties, map ) {  
  
    // Load some KML only into global maps - for instance pictures of squirrels  
  
    if (properties.map_content == 'global') {  
  
        // Make the Mapstraction call to add a Flickr KML layer  
        // This will only work with a provider that supports it (googlev3 does, openlayers  
        // doesn't)  
        map.addOverlay(  
            'http://api.flickr.com/services/feeds/geo/?g=52241987644@N01&lang=en-us&format=kml'  
        );  
  
    }  
  
} );
```

Or, you can bypass Mapstraction and use a provider's native API using the `Mapstraction.getMap` function. Here's a google V3 native example that might go in `custom-googlev3.js`:

```
// An Example Google V3 customization

GeoMashup.addAction( 'loadedMap', function ( properties, mxn ) {

    // Load some KML only into global maps - for instance pictures of squirrels

    var google_map = mxn.getMap();

    if (properties.map_content == 'global') {

        // Make the Google v3 call to add a Flickr KML layer
        var kml = new google.maps.KmlLayer(
'http://api.flickr.com/services/feeds/geo/?g=52241987644@N01&lang=en-us&format=kml',
{
    map: google_map
} );

    }

} );
```

All of the Geo Mashup methods and action events are listed in the [JavaScript API reference](#).

There are more examples in [user wiki guides](#). It can also be very useful to view the source of the custom javascript files of [the author's projects](#) and the user ExamplesInAction.

Location Editor API

The back end editor API is a little different in that it uses [Mapstraction events](#).

Here's an example of location-editor.js custom code that sets the center and zoom level of the editor map:

```
/* Specify the initial location/zoom level for the location editor map */  
  
geo_mashup_location_editor.mapCreated.addHandler( function() {  
    geo_mashup_location_editor.map.setCenterAndZoom( new mxn.LatLonPoint( 39,  
-119.1 ), 8 );  
} );
```

Methods and events are listed in the [JavaScript API reference](#).

The [generated documentation](#) covers all the classes and methods you can use.

Generated documentation doesn't include WordPress filters, so those are documented here along with some overviews and examples.

Geographical Queries

There are various ways to query by location, especially with posts.

Location query variables

A subset of the `TagReference#QueryVariables` are used for a few kinds of geographical queries.

Radius queries

A query to find all posts/objects within a distance from a search point is made using `near_lat`, `near_lng`, and `radius_km` or `radius_mi`.

Bounding box queries

A query to find all posts/objects within a bounding latitude and longitude range is made using minlat, maxlat, minlng, and maxlng.

Named area queries

A query to find posts/objects in a named area is made using one of:
admin_code(state/province), country_code, postal_code, or locality_name (city).

WP_Query Integration

Added in Geo Mashup 1.7.0.

Any of the location query variables can be used in a WP_Query post query with the geo_mashup_query variable. Here are some examples:

```
$nv_query = new WP_Query( array(
    'posts_per_page' => -1,
    'geo_mashup_query' => array(
        'admin_code' => 'NV',
    )
) );
```

```
$zip_query = new WP_Query( array(
    'posts_per_page' => -1,
    'geo_mashup_query' => array(
        'postal_code' => '96161',
    ),
) );
```

```
$radius_query = new WP_Query( array(
    'posts_per_page' => -1,
    'geo_mashup_query' => array(
        'near_lat' => 39,
```

```
'near_lng' => -120,
'radius_km' => 50,
),
) );
```

Any other [WP_Query parameters](#) can be added, allowing for powerful combinations.

The GM_Location_Query class

Added in Geo Mashup 1.7.0.

Geographical queries can be constructed for objects other than posts using this class, whose constructor takes an array of location query variables. An example of a user query:

```
$location_query = new GM_Location_Query( array(
    'minlat' => $nv_location->lat - 1,
    'maxlat' => $nv_location->lat + 120,
    'minlon' => $nv_location->lng - 1,
    'maxlon' => $nv_location->lng + 1,
) );
```

```
list( $cols, $join, $where, $groupby ) = $location_query->get_sql( $wpdb->users, 'ID' );
```

```
$sql = "SELECT {$wpdb->users}.ID FROM {$wpdb->users}{$join} WHERE
1=1{$where}";
```

Actions

Geo Mashup behavior can be augmented with callbacks in the form of [WordPress actions](#).

geo_mashup_init

Called when Geo Mashup has loaded and its API is available for use.

geo_mashup_render_map

Arguments:

- \$mashup_script - 'geo-mashup-mxn' or 'geo-mashup-google' for Google V2

Called when a map is being rendered and may be used to enqueue additional scripts.

Example

This is how a script is enqueued to add the blue dot to search results maps:

```
function prefix_geo_mashup_render_map() {  
  if ( 'search-results-map' == GeoMashupRenderMap::map_property( 'name' ) ) {  
    GeoMashup::register_script( 'geo-mashup-search-results', 'js/search-results.js', array(  
'geo-mashup' ), GEO_MASHUP_VERSION, true );  
    GeoMashupRenderMap::enqueue_script( 'geo-mashup-search-results' );  
  }  
}  
add_action( 'geo_mashup_render_map', 'prefix_geo_mashup_render_map' );
```

geo_mashup_added_object_location

Called when an object location is created.

Arguments:

- \$object_name - 'post', 'user', 'comment', etc.
- \$object_id
- \$geo_date
- \$location - the location object added

geo_mashup_added_updated_location

Called when an object location is updated.

Arguments:

- \$object_name - 'post', 'user', 'comment', etc.
- \$object_id
- \$geo_date
- \$location - the location object updated

geo_mashup_added_location

Called when a new location is added.

Arguments:

- location- the location added

geo_mashup_updated_location

Called when a new location is updated.

Arguments:

- location- the location updated

geo_mashup_deleted_object_location

Called when an object location is deleted.

Arguments:

- object_location- the object location deleted

geo_mashup_deleted_location

Called when an location is deleted.

Arguments:

- location- the object location deleted

Filters

Some Geo Mashup data can be altered using these [WordPress filters](#).

geo_mashup_locations_json_object

Allows you to change the object data that gets sent to a map. Called once for each object.

Arguments:

- \$json_properties - an associative array of properties to include in the object, including object_id, lat, lng, title, author, categories
- \$queried_object - the query result row returned for the object

Example

Use the saved name instead of the default object title (post_title for posts).

```
function my_geo_mashup_locations_json_filter( $json_properties, $queried_object ) {  
    $json_properties['title'] = $queried_object->saved_name;  
    return $json_properties;  
}  
add_filter( 'geo_mashup_locations_json_object', 'my_geo_mashup_locations_json_filter',  
10, 2 );
```


Now you can access the data in [custom javascript](#):

```
GeoMashup.addAction( 'objectIcon', function( properties, object ) {  
    // Use a special icon for the saved location 'Swimming Pool'  
    if ( 'Swimming Pool' == object.title ) {  
        object.icon.image = properties.template_url_path + 'images/pool_icon.png';  
    }  
} );
```

geo_mashup_locations_fields

Modify the fields included in a map query.

geo_mashup_locations_join

Modify the tables joined in a map query.

geo_mashup_locations_where

Modify the where clause of a map query.

geo_mashup_locations_orderby

Modify the orderby clause of a map query.

geo_mashup_locations_limits

Modify the limit clause of a map query.

geo_mashup_search_query_args

Modify the query arguments before a geo search is performed.

Arguments:

- \$query_args - array of [TagReference#Query_Variables](#).

geo_mashup_search_query_location_text

Modify the geo search text before it is geocoded.

Arguments:

- \$location_text - the location search string

Example:

```
/* add country to the geo mashup search */
add_filter( 'geo_mashup_search_query_location_text',
'my_geo_mashup_search_query_location_text_filter');

function my_geo_mashup_search_query_location_text_filter($location_text){
    if(!is_string($location_text) ) {return; }
    $location_text = $location_text.',DE';

    return $location_text;
}
```

geo_mashup_static_map

Modify the image HTML for static maps.

Arguments:

- \$map_image - the HTML tag. Return your filtered version.
 - \$map_data - An array of all data used to build the map.
 -
 - \$click_to_load - An array with click_to_load information:
 - 'click_to_load' - true if click_to_load is enabled



- 'click_to_load_text' - the text specified for the click_to_load link

Dependent Plugins

There are good possibilities for companion plugins to make use of Geo Mashup data.

If you need a Google API Key in your plugin, you can get Geo Mashup's with this PHP code: `$google_key = get_option('google_api_key');`. This setting is not deleted if Geo Mashup is uninstalled.