

# Google Static Maps V2 API

CodeIgniter Library

Author: BIOSTALL (Steve Marks)

Website: <http://biostall.com>

Link: <http://biostall.com/codeigniter-google-static-maps-v2-api-library>

Email: [info@biostall.com](mailto:info@biostall.com)

## Introduction

This CodeIgniter library provides an easy way to display simple static map images within an application/website using the Google Static Maps V2 API. It allows maps to be shown including customisable markers, paths and polygons with just a few lines of code.

## Installation

In order to use the library you will need to download the Googlemaps.php file and place it in your 'application/libraries' directory. Following the demonstrations and documentation below you will then be able to begin creating maps right away.

## The Basics

Once the installation instructions above are complete there are just two things we need to do in order to create a map image. The first is to amend our controller in order to initialize and customise our map image, and the second is to include a single line of code in our view where we want the map to be displayed:

### The Controller

The example below shows how to create a very simple standalone map:

```
// Load the library
$this->load->library('googlemaps');

// Initialize our map. Here you can also pass in additional parameters for customising the map (see below)
$config['center'] = '37.4419,-122.1419';
$config['zoom'] = 13;
$this->googlemaps->initialize($config);

// Create the map. This will return the image to be included in our pages where we want the map to appear.
$data['map'] = $this->googlemaps->create_map();

// Load our view, passing the map data that has just been created
$this->load->view('my_view', $data);
```

### The View

In the view simply echo the created output onto the page. In this example the variable is called \$map. This should be placed where you want the map image to appear as follows:

```
<?php echo $map; ?>
```

## Customising the Map

The library allows you to adjust the way your map appears by passing in additional values in a \$config array to \$this->googlemaps->initialize(\$config). These are as follows:

Name	Type	Default	Possible Values	Description
\$center	string		A latitude/longitude coordinate OR an address.	Sets the default center location (lat/long co-ordinate or address) of the map image. Required if markers not present. If blank and markers exist will default central position of all markers.
\$https	boolean	FALSE	TRUE, FALSE	If set to TRUE will load the API over HTTPS, allowing you to utilize the API within your HTTPS secure application
\$image_format	string	"png"	"png8", "png", "png32", "gif", "jpg", "jpg-baseline"	The image format output
\$include_img_tag	boolean	TRUE	TRUE, FALSE	If set to TRUE will include the full <img> tag. If FALSE just the image source will be returned
\$language	string			Defines the language to use for display of labels on map tiles. Note that this parameter is only supported for some country tiles
\$map_id	string	"map_canvas"		The ID of the image that is output containing the map image
\$map_height	integer	500		The height of the map image in pixels
\$map_type	string	"roadmap"	"roadmap", "satellite", "terrain", "hybrid"	The default MapType
\$map_width	integer	500		The width of the map image in pixels
\$sensor	boolean	FALSE	TRUE, FALSE	Set to TRUE if being used on a device that can detect a users location
\$zoom	string		0 (zoomed out) – 18 (zoomed in)	The default zoom level of the map image. Required if markers not present. If blank and markers exist will default zoom to include all markers

## Adding Markers

The library also allows you to add multiple markers to the map at specified positions. To add a single marker we can add the following code BEFORE the `create_map()` function is called:

```
// Set the marker parameters as an empty array. Especially important if we are using multiple markers
$marker = array();
```

```
// Specify an address or lat/long for where the marker should appear.
```

```
$marker[' position '] = 'Crescent Park, Palo Alto';
```

```
// Once all the marker parameters have been specified lets add the marker to our map
```

```
this->googlemaps->add_marker($marker);
```

To create multiple markers simply duplicate the above code the required amount of times.

Like the map itself, we can also specify a number of parameters for individual markers to change how and where they appear. These parameters are as follows:

Name	Type	Default	Possible Values	Description
\$position	string		A latitude/longitude coordinate OR an address.	The position at which the marker will appear
\$size	string		"tiny", "mid", "small"	The size of the icon to use for the marker. If no size is set the marker will appear in its default size.
\$color	string		A hex value (eg. #990000 or a word; black, brown, green, purple, yellow, blue, gray, orange, red, white)	The color of the icon to use for the marker
\$label	string		A-Z or 0-9	A single uppercase alphanumeric character to show on the markers icon. Tiny and small markers are not compatible with this option
\$icon_url	string			Instead of using the default marker icons you can specify a URL to a custom icon image. Limited to 4096 pixels and five unique custom icons per request
\$icon_shadow	boolean	TRUE	TRUE, FALSE	If using a custom icon this option turns on and off the shadow. If TRUE the shadow will be made based on the image's visible region and its opacity/transparency.

## Adding Paths

The library also allows you to add paths to the map at specified positions. To add a single path we can add the following code BEFORE the `create_map()` function is called:

```
// Set the path parameters as an empty array. Especially important if we are using multiple paths
$path = array();
```

```
// Specify an array of addresses or lat/longs for where the path points should appear.
$path['positions'] = array('37.429, -122.1319', 'Crescent Park, Palo Alto', '37.4419, -122.1219');
```

```
// Once all the path parameters have been specified lets add the path to our map
$this->googlemaps->add_path($path);
```

To create multiple paths simply duplicate the above code the required amount of times.

We can also specify a number of parameters for individual paths to change how and where they appear. These parameters are as follows:

Name	Type	Default	Possible Values	Description
<code>\$positions</code>	array		An array of latitude/longitude coordinates OR addresses, or a mixture of both.	Two or more positions at which the path will appear
<code>\$weight</code>	integer	5		The thickness of the path in pixels. Defaults to 5 pixels
<code>\$color</code>	string		A hex value (eg. #990000, or #FFFCCFF or a word; black, brown, green, purple, yellow, blue, gray, orange, red, white)	The color of the path
<code>\$fillcolor</code>	string		A hex value (eg. #990000, or #FFFCCFF or a word; black, brown, green, purple, yellow, blue, gray, orange, red, white)	Indicates the path is a polygonal area and specifies the fill color within that area. The array of locations positions need not be a "closed" loop; the Static Map server will automatically join the first and last points. Note, however, that any stroke on the exterior of the filled area will not be closed unless you specifically provide the same beginning and end location.

## Need Help?

To leave feedback, ask questions or report bugs please contact me at [info@biostall.com](mailto:info@biostall.com) or leave a comment at <http://biostall.com>.