Google Static Maps v2 plug-in

Namespace: `\Web\Google`
File location: `web/google/staticmap.php`

Class StaticMap

The Google Static Maps class provides a simple way to embed a Google Maps image on your web page without requiring JavaScript or any dynamic page loading.

Specifying Location

The Static Maps API uses strings (addresses) or numbers (latitude and longitude values) to specify locations on a map. (These values identify what is called a geocoded location)

To specify these locations, you can use either the `center` parameter and/or place any optional placemarks using the `markers` parameter at locations on the map.

As stated above, with both the `center` parameter and the `markers` parameter, you can use either Addresses or Latitudes and Longitudes to identify the location as shown in the examples below.

The **center** parameter

`center` defines the center of the map, equidistant from all edges of the map. This parameter takes a location as either a comma-separated {latitude,longitude} pair (e.g. "24.582720, -78.056685") or a string address (e.g. "Turks and Caicos Islands") identifying a unique location on the face of the earth

Note that addresses provided may reflect either precise locations, such as an accurate street address, polylines such as named routes, or polygonal areas such as cities, countries, or even national parks.

If for example you specify `$map->center('Grand Canyon');` as address, the Static Map server will resolve the location and return a map image using the center point of the area as the address center.

Examples:

```
$map = new \Web\Google\StaticMap();
// by area
$map->center('富士山'); // 'Mont Fuji' area
// by address
$map->center("D'Arblay Street, London"); // F3 will escape spaces, quotes, and specia
l characters for you
// by latitude and longitude values
$map->center('35.360496,138.727798'); // for example retrieved from a DB
```

The **markers** parameter

The `markers` parameter defines a set of one or more markers at a set of locations. Each marker descriptor must contain a set of one or more locations defining where to place the marker on the map. These locations are separated using the pipe character (|).

Example:

```
$map = new \Web\Google\StaticMap();
$map->markers('color:blue|label:S|San Francisco,CA'); // you can use an address or la
t,long coordinates
```

The plugin will automatically urlencode your settings, so don't worry about that. Check out the Google Static Maps API V2 docs (https://developers.google.com/maps/documentation/staticmaps/?hl=en#Markers) to see the full documentation about markers.

Images formats

Images may be returned in several common web graphics formats: GIF, JPEG and PNG. You can use the `format` parameter that takes one of the following values:

- `'png8'` or `'png'` (default) specifies the 8-bit PNG format.
- `'png32'` specifies the 32-bit PNG format.
- `'gif'` specifies the GIF format.
- `'jpg'` specifies the JPEG compression format.
- `'jpg-baseline'` specifies a non-progressive JPEG compression format.

Example:

```
$map = new \Web\Google\StaticMap();
$map->format('jpg'); // progressive "lossy" JPEG compression format
```

Maps Zoom level

Maps on Google Maps have an integer "zoom level" which defines the resolution of the current view.

Zoom levels between `0` (the lowest zoom level, in which the entire world can be seen on one map) to `21+` (down to individual buildings) are possible within the default roadmap maps view.

Note: Zoom levels vary depending on location and not all zoom levels appear at all locations on the earth.

Example:

```
$map = new \Web\Google\StaticMap();
$map->zoom(12); // resolution of the view
```

Images sizes

The `size` parameter, in conjunction with `center`, defines the coverage area of a map. It also defines the output size of the map in pixels, when multiplied with the `scale` value (which is 1 by default).

Unlike others, this parameter has no default value and you **must** specify it in your application otherwise the map server will reject your request and return an error message.

Example:

```
$map = new \Web\Google\StaticMap();
$map->size('640x480'); // in pixels and multiplied by scale
```

Scale Values

The `scale` value is multiplied with the `size` to determine the actual output size of the image in pixels, without changing the coverage area of the map. (Default `scale` value is 1; accepted values are 1, 2, and, for Maps API for Business customers only, 4).

Example:

```
$map = new \Web\Google\StaticMap();
$map->scale(2); // size multiplier
```

Notice: When targeting mobile devices, use the `scale` parameter to return higher-resolution map images that solve the issues of mobile devices high resolution screens (see the Google API Scale Values documentation (https://developers.google.com/maps/documentation/staticmaps/#scale_values) for more details).

Maps types

The Google Static Maps API creates maps in several formats, listed below:

- `roadmap` (default) specifies a standard roadmap image, as is normally shown on the Google Maps website. If no maptype value is specified, the Static Maps API serves roadmap tiles by default.
- `satellite` specifies a satellite image.
- `terrain` specifies a physical relief map image, showing terrain and vegetation.
- `hybrid` specifies a hybrid of the satellite and roadmap image, showing a transparent layer of major streets and place names on the satellite image.

Example:

```
$map->maptype( 'satellite' ); // Note: not all satellite views appear at all location
s on the earth.
```

Sensor

Use of the Google Static Maps API requires that you indicate whether your application is using a "sensor" (such as a GPS locator) to determine the user's location. This is especially important for mobile devices and Google decided to make it a mandatory parameter when requesting a static map. It means there is no default value and you **must** specify if your application is using a sensor device, otherwise the map server will reject your request and return an error message.

When set to `'true'`, on mobile devices, the Google Maps API will try to have a peek at the embedded GPS, while on a desktop browser, depending on the privacy settings, you might get a message saying the website try to determine your location.

If you're not sure about this "sensor", set it to `'false'` (as a string, not a boolean!).

```
$map->sensor( 'false' ); // 'true' or 'false' as a string, not a boolean!
```

Methods

dump

Generate map using the Google Static Maps API v2

```
string dump (  )
```

This method send a request to the Google Static Maps API service and returns the map as a raw image you can either save or display on your web page.

Example:

```
$map = new \Web\Google\StaticMap();

$map->center('Pulau Tomea'); $map->maptype('satellite');
$map->size('320x240'); $map->zoom('10');

// we'll save the map in TEMP for example
$map_filename = $f3->get('TEMP').'map.png'; // by default returned maps images are in
.png format

// save into a file the result of the API call retrieved by dump()
$f3->write($map_filename, $map->dump());

// use the image in the HTML page
printf ('<img src="/%s" alt="Map of Pulau Tomea in Lovely Indonesia" />', $map_filena
me);
```

There are a couple of drawbacks with this solution: you need to write a file to your server and, most importantly, the script has to wait the answer from the Google API to continue to parse and render your page. If for any reason the Google API is unavailable, your whole page will stuck at the very place the static map should appear.

Let's setup now an asynchronous solution. We need first to define a route for F3 to handle the request. Thus, the page can continue to be rendered while the browser send, in parallel, a request to this new route:

```
$f3->route('GET /static-map/@mapargs', function($f3, $args) {

    parse_str($args['mapargs']); // gets variables from the query string

    $map = new \Web\Google\StaticMap();

    $map->center(isset($center)?$center:'Center of the World');
    // override default maptype 'roadmap'
    $map->maptype(isset($maptype)?$maptype:'satellite');
    // override default format 'png'
    $map->format($format=(isset($format)?$format:'jpg'));
    // size is mandatory! no default value. set default size '320x200'
    $map->size(isset($size)?$size:'320x200');
    // override default zoom '12'
    $map->zoom(isset($zoom)?$zoom:'14');
    // override default scale '1'
    $map->scale(isset($scale)?$scale:'2');
    // sensor is mandatory! no default value. set sensor to 'false'
    $map->sensor(isset($sensor)?$sensor:'false');

    // send raw image map to browser
    header('Content-type: image/'.$format);
    echo $map->dump();
},
    3600*24*30 // Save result of the request in F3 cache for 30 days
);
```

We are ready to asynchronously retrieve a static map from the Google API:

```
echo '<img src="/static-map/size=640x480&scale=1&center=Mont%20Fuji" />';
```

or, much probably, as you would do in a template:

```
<img src="/static-map/maptype=hybrid&center={{@MAP.center}}" alt="{{@MAP.title}}" />
```

Map of Pulau Tomea in Indonesia

You can do more with the Google Static Maps API, especially regarding Markers (https://developers.google.com/maps/documentation/staticmaps#Markers) and Styled Maps (https://developers.google.com/maps/documentation/staticmaps#StyledMaps). Please refer to the official Google Static Maps API V2 Developer Guide (https://developers.google.com/maps/documentation/staticmaps/) for more details about advanced features.