EXT: WEC Map

Extension Key: **wec_map**

Author: Web Empowered Church Team, <map@webempoweredchurch.org>

Support Community: http://www.webempoweredchurch.com/community/

The content of this document is related to TYPO3
- a GNU/GPL CMS/Framework available from www.typo3.com

Table of Contents

Introduction

A. Welcome

Welcome to the adventure of doing ministry through technology!

This manual is written for those who are interested in using Web-Empowered Church (WEC) technology and enhancing their website through the use of the WEC Map extension. The WEC Maps extension has been designed to provide churches and ministries an opportunity to display maps on their website...maps to the church, maps to events, even maps that show how many members live in the surrounding towns.

What is the Web-Empowered Church?

The Web-Empowered Church (WEC) is a software ministry of the Foundation for Evangelism (www.evangelize.org). The mission of WEC is to innovatively apply WEB technology to EMPOWER the worldwide CHURCH for ministry.

WEC helps churches around the world expand evangelism, discipleship, and care through the innovative application of internet technology. WEC web-based tools and training will help make church ministries more efficient and effective, and will extend ministry impact to a world in need of Jesus (www.webempoweredchurch.org/Jesus). We want to fuel a worldwide movement using the internet to point the world to Jesus Christ, to grow disciples, and to care for those in need. Our desire is to use the web to empower the Church to become a truly twenty-four hours a day, seven days a week ministry that is not constrained by walls or distance or time.

If you would like to find out more about WEC or our tools, or support us in any way, please go to our websites:

for users: www.webempoweredchurch.com

for developers: www.webempoweredchurch.org.

B. About this Manual

This guide is organized into four main sections:

I. **Overview:** What the WEC Map extension does, how it works, and what you can expect from the rest of the manual.

II. **Installation:** How to install the WEC Map extension and how to get it running on your website.

III. **Configuration:** How to configure the WEC Map extension and plugins for maximum impact on your website.

IV. **Advanced:** How to modify the extension through TypoScript and other advanced functions.

C. How to Use this Manual

✔ To help make sense of the material on the following pages, we've used a few icons in the margin to highlight certain information.

Tip: Tips are bits of information that are good to know. They may offer shortcuts to save you time or even make your website better.

Note: These notes are similar to tips, but usually contain information you should pay attention to. It might be details about a step that a whole operation hinges on or it may highlight an essential sequence of tasks.

Caution: These notes draw your attention to things that can interrupt your service or website if not done correctly. Some actions can be difficult to undo.

Technical Stuff: These notes will explain how stuff works. If you want to know why certain steps are necessary or are just looking for a fuller explanation, these notes should help answer your questions.

Internet Link: These links will guide you to a website page that will demonstrate How-To do something or link you to more advanced information on the http://webempoweredchurch.com or http://typo3.org websites.

Overview

A. What does it do?

The Web-Empowered Church Map Extension provides an interface for placing maps from Google on a TYPO3-based website. There are two actual plugins provided with the WEC Map extension.

1. **The Simple Map Plugin** allows you to show a single address, such as your church address, on a Google map. See Illustration 1. However, by adding TypoScript to configure the WEC Map extension, you can add multiple addresses and multiple markers (see Marker Manager).
2. **The Frontend Users Map Plugin** allows you to show multiple locations on a map. These locations are drawn from a list of Frontend Users and can be limited to a particular Frontend Users Group. See Illustration 2. This provides you with the opportunity to map the location of off-site small groups, multiple site campuses, or even member addresses. A security option is available so that names and addresses can be hidden if desired. When the security option is enabled, only the city, state, and country for each user is located on the map to hide the specific locations of Frontend User addresses.

B. Screenshots

Illustration 1: Simple Map

Illustration 2: Frontend Users Map

Installation

There are four steps necessary to install the WEC Map extension and plugin onto your website. These steps include:

1. Importing and installing the WEC Map extension.
2. Adding the Google Map API Key to the extension.
3. Adding the WEC Map Static Template to your page template.
4. Adding the WEC Map plugin to a web page.

Step 1: Import and Install the wec_map Extension

To import and install the WEC Map extension, you must be logged in to your website's TYPO3 Backend and have administrative privileges.

Full instructions for uploading and installing extensions can be found in the How-To located on the Web-Empowered Church website: http://webempoweredchurch.com/support/howtos/. (Use **wec_map** in your repository search).

Step 2: Add the Google Map API Key to the Extension

Before you can use the WEC Map extension, you must obtain a unique Google Map API Key for your website and install it either into the extension or into the plugin. We suggest adding the API Key to the extension so that the WEC Map extension is available and functional throughout the website (globally).

Technical Stuff: New users may become confused by the two terms Extension and Plugin, but they refer to two very different "parts" of the TYPO3 experience.

- ***Extension:*** An extension is like a sub-program. Extensions "extend" the capabilities of TYPO3. Most installed Frontend extensions, however, reside silently in the Backend and don't "work" until you have added the extension's plugin as a Content Element.
- ***Plugin:*** A plugin is the Frontend application of an extension. It's what makes

the extension do what it's supposed to do. Plugins must be applied to a page as a Content Element. A single extensions may have several plugins.

1. Refer to Illustration 3. Open an Internet browser and go to www.google.com/apis/maps (1) to begin the process of getting a Google Map API Key.
2. Click on the link to **Sign up for a Google Map API Key** (2).

Illustration 3: Obtaining a Google Map API Key 1

3. Refer to Illustration 4. Read the disclaimer and type the URL of your website (3). Click the **Generate API Key** button (4).

Illustration 4: Obtaining a Google Map API Key 2

4. Refer to Illustration 5. Your API Key will be generated and available in the text box (5). Highlight the complete key (it may be necessary to triple click the letters of the key itself) and copy the key to your clipboard (Control-C).

Illustration 5: Obtaining a Google API Key 3

5. Refer to Illustration 6. Return to the TYPO3 Backend. Using the **WEC Map Admin Module** (6), select **API Key Settings** from the dropdown box (7), and enter your API key in the text field under the right suggested domain. If the correct domain is not suggested, click on "Manually add a new API key for domain"(8), then enter the domain and key in the form that appears. Lastly, hit submit to save the changes.

Illustration 6: Adding the API Key to the Extension 1

Step 3: Add the WEC Map Static Template

In order for the WEC Map extension to display, the WEC Map Static Template should be added to the page template where the Map extension is installed. For most websites, this will mean adding the Static Template to the Root Page; however, if you have created a unique page template for the map page, you may need to add the Static Template there instead.

1. Refer to Illustration 8. Using the **Template Module** (1), click on the page where the page template is located--typically, the **Root** page (2). In the far right drop-down box, select **Info/Modify** (3). Click the **Click here to edit the whole template record** link (4).

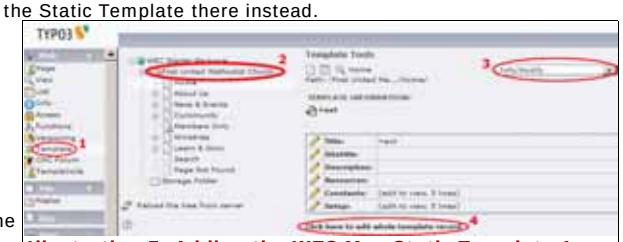

Illustration 7: Adding the WEC Map Static Template 1

2. Refer to Illustration 9. Scroll down the template's form to the **Include Static** Field. Scroll down the **Items** list and select either or both of the Map's Static Templates: **WEC Frontend User Map** and/or **WEC Simple Map depending on the plugin(s) you intend to use**.

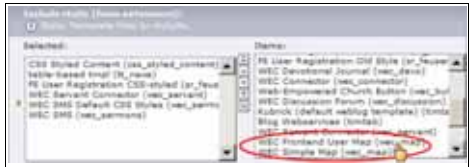

Illustration 8: Adding the WEC Map Static Template 2

3. Save and close the form.

Step 4: Add the WEC Map Plugin to a Page

The WEC Map extension can be added to any page as a standard **Content Element**. However, at this time there can only be one instance of the Map plugin on any single page. If you need to show multiple maps we suggest adding additional pages with one map per page.

To add the WEC Map plugin to a page:

1. Refer to Illustration 10. Using the **Page Module** (1), select the page from the **page tree** where you'd like to place the map (2).
2. Add a **Content Element** in the Content Container of your choice (3).

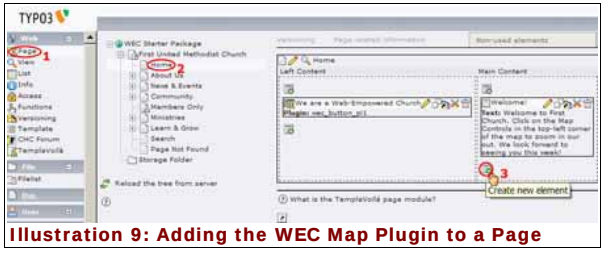

Illustration 9: Adding the WEC Map Plugin to a Page

Tip: You can add the WEC Map plugin, or any Content Element to a Content Container by selecting the Create New Element icon. These icons will appear wherever you are allowed to add content such as images, text, or additional plugins.

3. Refer to Illustration 11. Scroll down the content form and select which of the two WEC Map plugins you'd like to use (4).

4. This will complete the installation of the WEC Map plugin on your page. The WEC Map FlexForm that will appear corresponds to either the WEC Simple Map Configuration or the WEC Frontend User Map Configuration.

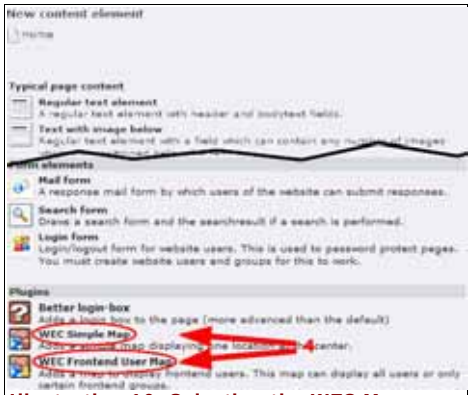

Illustration 10: Selecting the WEC Map Plugin

Configuration

Once you've installed one of the plugins on your page, you will need to configure the plugin so that it will display the map of your choice. Skip to the instructions that correspond to the WEC Map plugin installed on your page.

Configuring the WEC Simple Map Plugin

The WEC Simple Map plugin is used to display a single location on the map such as your church or ministry address. To configure the plugin, use the **Page Module** and select the page where you've installed the plugin. Then click the Edit Icon (the pencil) associated with the WEC Simple Map plugin. This will open the plugin's FlexForm.

Tip: When working with a plugin, it is generally helpful to write the plugin's name or some other descriptive title in the **Header** field (1). This makes it easier to select which plugin you want to edit when a page contains multiple **Plugin Content Elements**. To prevent the header from displaying on the page, change the **Header Type** to **Hidden** (2).

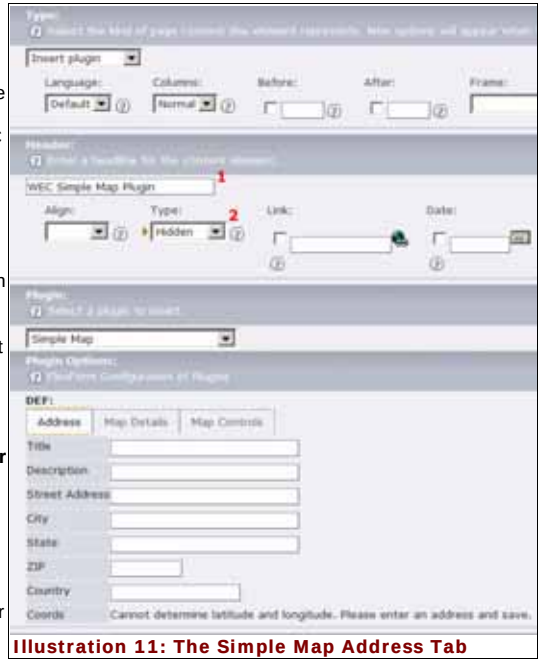

Illustration 11: The Simple Map Address Tab

1. Refer to Illustration 12. Note that there are three Tabs in the **Plugin Options** form. The first tab, labeled “Address,” prompts you for a description of the address to be mapped, as well as the address itself. It will also show the status of finding the coordinates for the address entered (geocoding) at the end.

2. Refer to Illustration 13. The second tab, labeled “Map Details,” requests display information about the map, such as the height and width. In addition, if you have not installed the Google API Key in the extension (Step 2: Add the Google Map API Key to the Extension), you will need to paste the API Key here or the map will not display.

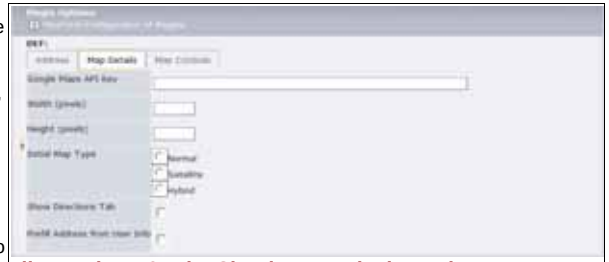

Illustration 12: The Simple Map Display Tab

The map's default height and width are both 500px, but can be changed in this form. Here you can also specify which map type should be initially shown.

One of the enhanced features of the Map extension is that if you enable the **Show Directions Tab**, Frontend users will have the option to be given directions to and from the location shown on the map. In addition, if they are a registered Frontend user and have logged in, if you have enabled the **Prefill Address from User Info**, their address will be automatically applied to the directions fields and the user can immediately click for Directions To or From the mapped location.

Caution: In order for the marker balloon information to be visible, avoid setting the height value at less than 400 pixels.

3. Refer to Illustration 14. The third tab, labeled “Map Controls,” allows you to specify which map controls should be shown on the map. Options include the size of the map controls (pan and zoom) if any, whether an overview map is desired, whether a scale should be included, and the map type selection control (satellite, political, or a hybrid of the two). See Illustration 15 for a visual description of these controls.

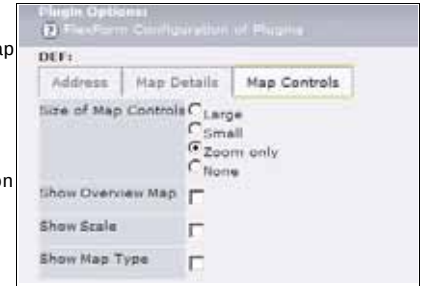

Illustration 13: The Simple Map Controls Tab

After saving this form and viewing its page in the Frontend, a map will be generated similar to Illustration 1, depending on the Map Controls you have chosen to display.

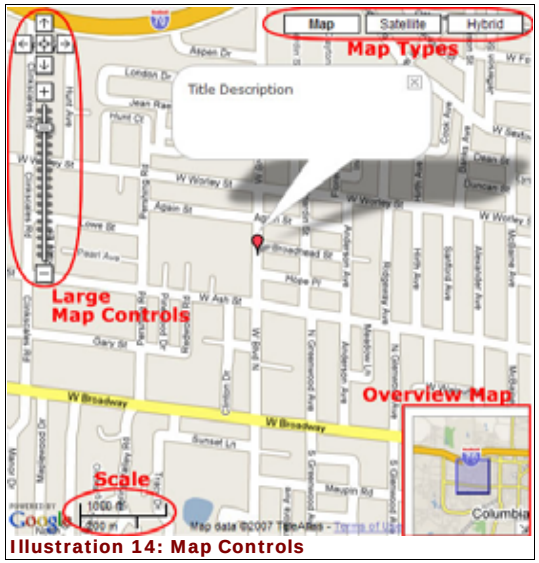

Illustration 14: Map Controls

Configuring the WEC Frontend User Map Plugin

The WEC Frontend User Map plugin is used to display a map of all users or all users belonging to one or more Frontend Users Groups. To open the plugin, use the **Page Module** and select the page where you've installed the plugin. Then click the Edit Icon (the pencil) associated with the WEC Frontend User Map plugin. This will open the plugin's FlexForm.

> **Tip:** When working with a plugin, it is generally helpful to write the plugin's name or a descriptive title in the **Header** field. This makes it easier to select which plugin you want to edit when a page contains multiple **Plugin Content Elements**. To prevent the header from displaying on the page, change the **Header Type** to **Hidden**. (See Illustration 12)

1. Refer to Illustration 16. Note that there are two tabs The first tab, labeled "Map Details" prompts you for the following:

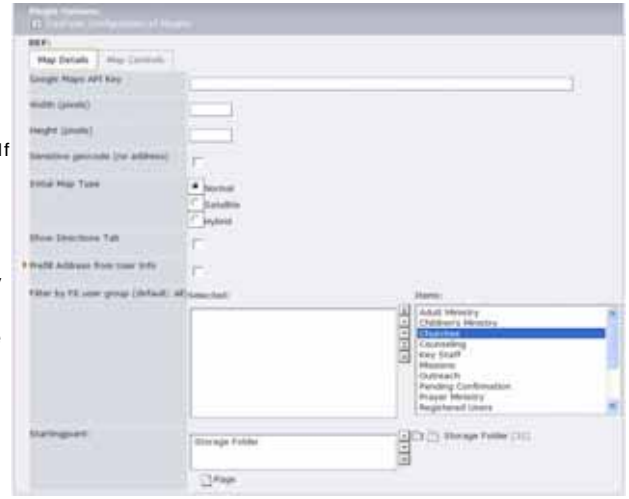

Illustration 15: The Frontend User Map Details Tab

- **Google Map API Key:** If you have not installed the API Key in the extension (Step 2: Add the Google Map API Key to the Extension), you will need to obtain a key and insert it here.
- **Width and Height:** The default size of the Frontend map is 500px by 500px. You can change the map size to suit the look and feel of your website here. In order for the marker balloon information to be visible in the Frontend, avoid setting the height value below 400 pixels.
- **Sensitive geocode (no address):** Checking this box prevents the map markers from displaying at the precise location of the user address and hides the detailed address information. Instead, Frontend users are shown by city, state, and country; and multiple users in the same city are totaled. (See Illustration 17 for examples of each.)

Illustration 16: Frontend Users Map With and Without Sensitive Geocoding

- **Initial Map Type:** Specifies which map type to show initially. "Normal" will show the street map (default), Satellite will show satellite images, and Hybrid will show satellite images with streets overlayed.
- **Show Directions Tab:** Displays an addition tab in the marker popup with two forms to get direction from or to the current location via maps.google.com. Directions are disabled if "Sensitive geocode" is enabled above, no matter the setting here.
- **Prefill Address from User Info:** If directions are enabled, this will automatically fill in a logged in FE user's address information into the directions form.
- **Filter by FE user group:** This is the key component to the Frontend Users Map plugin. Select the Frontend User Group(s) you would like to include on your map. You may create custom Users and User Groups as necessary to display the locations of a multi-site campus, locations of local churches, small groups, etc. (For information on how to add Frontend Users, see www.webempoweredchurch.com/support/howtos/).

Refer to Illustration 18. The third tab, labeled "Map Controls," allows you to specify which map controls should be shown on the map. Options include the size of the map controls (pan and zoom) if any, whether an overview map is desired, whether a scale should be included, and the map type selection control (satellite, political, or a hybrid of the two). See Illustration 15 for a visual description of these controls.

After saving this form and viewing its page in the Frontend, a map will be generated in a similar to Illustration 2, depending on the Map Controls you have chosen to display.

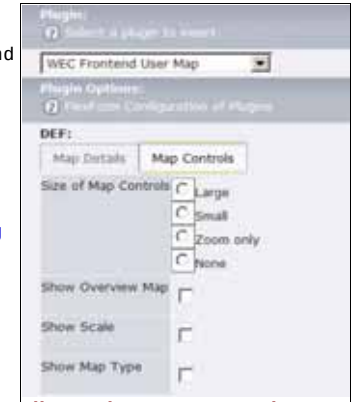

Illustration 17: Frontend User Map Controls Tab

Advanced Configuration with TypoScript

The WEC Map extension can be configured using standard TypoScript and FlexForms. Example TypoScript is provided below for both plugins. Note that for the WEC Simple Map, which is referred to as "tx_wecmap_pi1" in the TypoScript, you can either specify the address in parts or as one string. By using TypoScript to configure this plugin you can also set more than one marker to appear on the map. The WEC Frontend User Map extension is referred to as "tx_wecmap_pi2."

```
plugin.tx_wecmap_pi1 {
      apiKey =
      height = 500
      width = 500
      showDirections = 1
      prefillAddress = 0
      initialMapType = G_SATELLITE_MAP
      controls.mapControlSize = small
      controls.showOverviewMap = 1
      controls.showMapType = 1
      controls.showScale = 1
      markers.1 {
         title = TS title
         description = TS desc
         street = 1234 Happy Place
         city = Happy City
         zip = 12345
         state = HS
         country = Happy Country
      }
      markers.2 {
            title = Another title
            description = Another description
            address = 1234 Another Happy Ave, Happy City, HS 12347, Happy Country
```

```
        }
        markers.3 {
                #add another marker here
        }
}

plugin.tx_wecmap_pi2 {
        apiKey =
        height = 500
        width = 500
        showDirections = 1
        prefillAddress = 0
        initialMapType = G_HYBRID_MAP
        controls.mapControlSize = large
        controls.showOverviewMap = 1
        controls.showMapType = 1
        controls.showScale = 1
        userGroups = 2,3,5
        pid = 2,3,5
}
```


Simple Map TypoScript Reference

Property	Data Type	Description	Default
apiKey	String	The Google Maps API key	
height	integer	The height of the map in pixels. Avoid values of less than 400.	500
width	integer	The width of the map in pixels	500
showDirections	boolean	Toggle whether a directions tab should be shown for map markers. If set to 1, the user can get directions from and to the current marker via maps.google.com directions.	0
prefillAddress	boolean	If enabled and a FE user is currently logged in, his or her address will automatically be filled into the directions form.	0
initialMapType	String	Specifies whether to show the street map, satellite images, or hybrid view initially. Possible values are the constants defined by Google: G_NORMAL_MAP for the normal street map, G_SATELLITE_MAP for Google Earth satellite images, and G_HYBRID_MAP for the hybrid view.	
controls.mapControlSize	String	Defines the map controls in the top left corner of the map. Possible values are “small”, “large”, or “zoomonly.” “Small” will only show the pan controls with a + and – zoom, “large” will show the pan controls with a zoom scale, and “zoomonly” will only show + and – zoom buttons.	
controls.showOverviewMap	boolean	Toggles the little mini map in the bottom right corner	0
controls.showMapType	boolean	Toggles the map type selectors (Map, Satellite and Hybrid) in the top right corner.	0
controls.showScale	boolean	Toggles the scale shown in the bottom left.	0
markers.*.title	String	Sets the title for a marker. * is a number starting at 1, which makes it possible to define as many markers as desired.	
markers.*.description	String	Sets the description for the marker.	
markers.*.street	String	The street name for the marker, i.e. 1234 Happy Lane	
markers.*.zip	String	The ZIP code for the marker, i.e. 43234	
markers.*.city	String	The city for the marker, i.e. Some City	
markers.*.state	String	The state for the marker, i.e. Montana	
markers.*.country	String	The country for the markers, i.e. USA	
markers.*.address	String	A whole address string. This can be used instead of the above 5 properties to define the address in one single string, i.e. 1234 Happy Lane, Some City, Montana 43234, USA	
zoomLevel	int	Sets the initial zoom level for the map. Values range from 0 (zoomed out all the way) to 17 (zoomed in all the way)	
centerLat	float	Set the latitude for the initial center of the map	
centerLong	float	Set the longitude for the initial center of the map	
mapName	String	Identifies the map uniquely on a page by setting it as the id of the div containing the map. It's also appended to all the javascript variables and functions so as not to conflict with more than one map on a page. Only necessary in special cases, usually the defaults work well enough.	mapXXX, where XXX is the content element's uid or, if not available, a random number.

FE User Map TypoScript Reference

Property	Data Type	Description	Default
apiKey	String	The Google Maps API key	
height	integer	The height of the map in pixels	500

width	integer	The width of the map in pixels	500
showDirections	boolean	Toggle whether a directions tab should be shown for map markers. If set to 1, the user can get directions from and to the current marker via maps.google.com directions.	0
prefillAddress	boolean	If enabled and a FE user is currently logged in, his or her address will automatically be filled into the directions form.	0
controls.mapControlSize	String	Defines the map controls in the top left corner of the map. Possible values are “small”, “large”, or “zoomonly.” “Small” will only show the pan controls with a + and – zoom, “large” will show the pan controls with a zoom scale, and “zoomonly” will only show + and – zoom buttons.	
controls.showOverviewMap	boolean	Toggles the little mini map in the bottom right corner	0
controls.showMapType	boolean	Toggles the map type selectors (Map, Satellite and Hybrid) in the top right corner.	0
controls.showScale	boolean	Toggles the scale shown in the bottom left.	0
userGroups	integer list	A comma separated list of FE user group ids. If you would like to restrict the markers shown to users from certain FE user groups, add their ids here. I.e., to only show users from FE user groups with ids 3 and 4, enter “3,4.”	
pid	integer list	A comma separated list of storage folder ids. If you would like to restrict the markers shown to users from only certain storage folders, add their ids here. I.e., to only show users from storage folders with pids 5 and 8, enter “5,8.”	
zoomLevel	int	Sets the initial zoom level for the map. Values range from 0 (zoomed out all the way) to 17 (zoomed in all the way)	
centerLat	float	Set the latitude for the initial center of the map	
centerLong	float	Set the longitude for the initial center of the map	
mapName	String	Identifies the map uniquely on a page by setting it as the id of the div containing the map. It’s also appended to all the javascript variables and functions so as not to conflict with more than one map on a page. Only necessary in special cases, usually the defaults work well enough.	mapXXX, where XXX is the content element’s uid or, if not available, a random number.

See the Installation section for more detailed information about configuring the two plugins via FlexForms.

Troubleshooting

1. In Internet Explorer 7 (IE7), when using the FE User Map plugin in the Backend, occasionally a slow server connection will cause an error when adding a new Frontend User in your Storage Folder.

   Internet Explorer cannot open the site http://www.yoursite.com/typo3/alt_main Operation aborted.

   To work around this issue, open the **Ext Manager Module**, and ensure you can view **Loaded extensions**. Scroll down to the **WEC Map** extension and click on the WEC Map link (the words). Uncheck the **Enable map in FE user records** and then click the **Update button**.

2. Some users who have large numbers of Frontend User records may discover that many of their users do not receive geocoding from Google. This issue may be caused because of your host server's firewall that interprets multiple geocoding requests and replies from Google.com as a Denial of Service Attack.

   To work around this issue, contact your host and ask them to **whitelist google.com** for your domain.

3. Sometimes the **marker bubbles appear to be empty** or appear to be missing either a title or a description. The reason for that is the font color used for headlines and body text across the whole website. If either is set to white font color, it will also render the headlines and/or body text white, making the marker bubbles appear empty. To fix it, make sure you **change the CSS style for the map element fonts**; it uses the class “tx-wecmap-map”, so you could add the following to your CSS styles file:

```
.tx-wecmap-map {color: black;}
.tx-wecmap-map p {color: black;}
```


Administration

Nested within the Subsection Tools Directory, there are two Backend Modules associated with the WEC Map extension.

WEC Map Admin Module

See Illustration 19. The WEC Map Admin Module (**1**) includes three tools in the dropdown list (**2**):

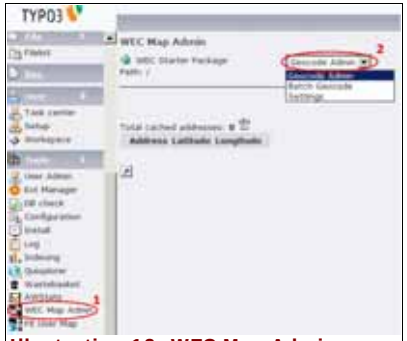

Illustration 18: WEC Map Admin Module

1. **Geocode Admin:** Displays a list of cached (stored) addresses with their longitude and latitude. Using the pencil icon, the latitude and longitude values can be changed manually. You can also use the Trashcan icon next to each entry to delete the entry completely. If there is a need, it will be automatically regenerated the next time the latitude and longitude is needed for a map. These coordinates come from Google or other geocoding services and are stored locally to decrease the display time of the map. A search box at the top allows the list of caches to be filtered by keyword and the record table can now be sorted by clicking on the table headers.
2. **Batch Geocode:** Manually initiates processing of assigning latitude and longitude to any records that support geocoding, including website users. This feature is particularly helpful when there are a significant number of users to process and the WEC Map extension was recently installed. You do not need to run the batch geocode because the WEC Map extension will automatically lookup any needed addresses, but using this feature will avoid long delays in the map display caused by many new geocode lookups. If you have over 100 addresses that must be geocoded, then the processing could take several minutes, but once an address is processed it will not need to be geocoded again unless the address changes.
3. **API Key Settings:** Allows you to insert a Google Map API Key directly into the extension to allow global access based on domain. This is useful if you are running a site with different domains. Once you have set API keys for each of your domains, the extension will automatically use the right one depending on the domain the user is visiting or logged in from.

FE User Map Module

See Illustration 20. The FE User Map Module (**1**) shows a detailed map of all the Frontend Users and is an example of how to use the API in backend development. It works very similar to the Frontend User Map plugin. Zooming in on a user and toggling the information bubble will display the address information for that user along with a link to edit the record directly (**2**).

Illustration 19: FE User Map Module

Using the WEC Map API

Two Frontend plugins are included with this extension, but other extensions can use the underlying API to provide various map functions as well.

Create the Map Object

The first step in using the WEC Map API is to create a map object. To do that, include the Google Maps class in your code and create an instance of it, passing the API key and the map's height and width as parameters. If the API Key was set through the extension manager, the value can be left empty.

```
include_once(t3lib_extMgm::extPath('wec_map').'map_service/google/class.tx_wecmap_map_google.php');
$className = t3lib_div::makeInstanceClassName("tx_wecmap_map_google");
$map = new $className($apiKey, $width, $height);
```

Configure the Controls

The next step is to configure the map controls. This is done by the *addControl* method of the map object. To fully load the map with all available controls, use the code below:

```
$map->addControl('largeMap');
$map->addControl('smallMap');
$map->addControl('smallZoom');
$map->addControl('scale');
$map->addControl('overviewMap');
$map->addControl('mapType');
```

The naming of the control strings passed to the *addControl* method coincides with Google's naming of the control, so 'smallMap' creates Google's GSmallMapControl, 'scale' creates Google's GScaleControl, etc.

Other Options

Another option that can be set is the initial map type, i.e. whether to show the street map, the satellite images, or the hybrid view. To do that, just call the *setType()* method on the map object.

```
$map->setType('TYPE');
```

TYPE can be any Google Type Constants specified in the API reference for GmapType (http://www.google.com/apis/maps/documentation/reference.html#GMapType). Possible values at this time are G_NORMAL_MAP for the normal street map, G_SATELLITE_MAP for Google Earth satellite images, and G_HYBRID_MAP, a mix of the previous two.

If this method is not used, the map will default to the G_NORMAL_MAP type.

To enable the directions tab to provide directions to or from a location using maps.google.com, use the *enableDirections()* method. It takes two optional parameters. The first is a boolean to toggle whether or not a logged in FE user's address should be automatically filled in to the directions form of the marker tab.

```
$map->enableDirections(); // enable directions tab, but don't automatically fill in user's address
$map->enableDirections(true); // enable directions and automatically fill in user's address
```

The second is a string with the id of a container element that will hold the written out driving instructions. This is *null* by default, which means no written out driving instructions will be shown. Make sure the id is unique on the page if you want to support more than one map on a page.

```
// enable directions, prefill user addresses, and show the written out driving instructions in the
//container with the id "directions_div".
$map->enableDirections(true, "directions_div");
```

Create Markers

Available methods

There are several methods that can be used to create markers on the map.

Method	Parameters	Description
addMarkerByAddress	$street	The street part of the address, e.g., 1234 Memorial Lane
	$city	The city part of the address, e.g., Oklahoma City
	$state	The state or region, e.g., Texas
	$zip	The zip code, e.g., 41343
	$country	The name of the country
	$title = ''	The title that will be displayed in the popup balloon that appears when this marker is clicked. **Default:** empty String
	$description = ''	The description that will be displayed in the popup balloon that appears when this marker is clicked. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** empty String
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Method	Parameters	Description
addMarkerByLatLong	$lat	The latitude coordinate for the marker
	$long	The longitude coordinate for the marker
	$title = ''	The title that will be displayed in the popup balloon that appears when this marker is clicked. **Default:** empty String
	$description = ''	The description that will be displayed in the popup balloon that appears when this marker is clicked. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** empty String
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Method	Parameters	Description
addMarkerByString	$string	The complete address as a single string
	$title = ''	The title that will be displayed in the popup balloon that appears when this marker is clicked. **Default:** empty String
	$description = ''	The description that will be displayed in the popup balloon that appears when this marker is clicked. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** empty String
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17
addMarkerByTCA	$table	Name of the table that contains address data
	$uid	The uid of the record to be mapped
	$title = ''	The title that will be displayed in the popup balloon that appears when this marker is clicked. **Default:** empty String
	$description = ''	The description that will be displayed in the popup balloon that appears when this marker is clicked. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** empty String
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Method	Parameters	Description
addMarkerByLatLongWithTabs	$lat	The latitude coordinate for the marker
	$long	The longitude coordinate for the marker
	$tabLabels = null	An array of strings that will be used to label the tabs. $tabLabels[0] will be the label of the first tab, etc. **Default:** null
	$title = null	An array of strings that will be used as titles in the different tabs. $title[0] will be used in the first tab, $title[1] in the second, etc. Default: null
	$description = null	An array of descriptions that will be displayed in the popup balloon that appears when this marker is clicked. $description[0] will be the description used in the first tab, $description[1] will be used in the second tab, etc. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** null
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Method	Parameters	Description

addMarkerByAddressWithTabs	$street	The street part of the address, e.g., 1234 Memorial Lane
	$city	The city part of the address, e.g., Oklahoma City
	$state	The state or region, e.g., Texas
	$zip	The zip code, e.g., 41343
	$country	The name of the country
	$tabLabels = null	An array of strings that will be used to label the tabs. $tabLabels[0] will be the label of the first tab, etc. **Default:** null
	$title = null	An array of strings that will be used as titles in the different tabs. $title[0] will be used in the first tab, $title[1] in the second, etc. **Default:** null
	$description = null	An array of descriptions that will be displayed in the popup balloon that appears when this marker is clicked. $description[0] will be the description used in the first tab, $description[1] will be used in the second tab, etc. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** null
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Method	Parameters	Description
addMarkerByStringWithTabs	$string	The complete address as a single string
	$tabLabels = null	An array of strings that will be used to label the tabs. $tabLabels[0] will be the label of the first tab, etc. **Default:** null
	$title = null	An array of strings that will be used as titles in the different tabs. $title[0] will be used in the first tab, $title[1] in the second, etc. **Default**: null
	$description = null	An array of descriptions that will be displayed in the popup balloon that appears when this marker is clicked. $description[0] will be the description used in the first tab, $description[1] will be used in the second tab, etc. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** null
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

addMarkerByTCAWithTabs	$table	
	$uid	
	$tabLables	An array of strings that will be used to label the tabs. $tabLabels[0] will be the label of the first tab, etc. **Default:** null
	$title = null	An array of strings that will be used as titles in the different tabs. $title[0] will be used in the first tab, $title[1] in the second, etc. **Default**: null
	$description = null	An array of descriptions that will be displayed in the popup balloon that appears when this marker is clicked. $description[0] will be the description used in the first tab, $description[1] will be used in the second tab, etc. Note that Google Maps is picky about line breaks in the title and description. The API will automatically replace line breaks with , so if you need your title and description to be processed differently, make sure to do that in your code before you pass it on to this method. **Default:** null
	$minzoom = 0	The minimum zoom level that this marker is supposed to appear on. The default is 0, which is the most zoomed out level. See notes on the Marker Manager below for more information on this. **Default:** 0
	$maxzoom = 17	The maximum zoom level that this marker is supposed to appear on. The default is 17, which is the most zoomed in level. See notes on the Marker Manager below for more information on this. **Default:** 17

Marker Manager

The Marker Manager is Google's answer to displaying multiple markers on the map at once, while at the same time keeping loading times low and speed high. The Marker Manager ensures that only those markers that are visible in the viewport are actually rendered. It also adds a "padding" of markers that are not yet visible but right outside the viewport to speed up marker rendering when the map is moved around.

The $minzoom and $maxzoom parameters can be used to specify on which zoom levels a certain group of markers should appear. The FE User Map plugin and the Backend module use this technique to show only a few markers per country on zoom levels 0 through 2, only one marker per city on zoom levels 3 through 7, and all individual markers on levels 8 through 17. See the plugin code and Google's Map API Documentation for more details.

Drawing the Map

Rendering of the map is simple. Just call the *drawMap()* method on the Map object and it will return the HTML and Javascript necessary to display the map within a Frontend Plugin or Backend Module:

```
$output = $map->drawMap();
```

Change Log

Version	Changes
1.3.6	[FIXED] Regression bug: Filtering the FE Map by usergroup now works correctly if users are in more than one group
1.3.5	[FIXED] wrong function call in the Batch Geocoder
1.3.4	[FIXED] Google Geocoder works correctly again
1.3.3	[FIXED] Filtering the FE Map by usergroup now works correctly if users are in more than one group

1.3.2	[NEW] Localize Google Maps error messages
	[NEW] Add a conflicting extension
	[FIXED] Bug when addresses are normalized in some places (_cache) and not in others (_backend)
	[FIXED] Incorrect API key usage when geocoding from the backend
	[FIXED] Add more descriptive error message for directions
	[FIXED] Update docs with 1.3.1 changelog and fix wrong CSS reference
	[FIXED] Map now zooms properly when only a manual center is given
1.3.1	[NEW] Add German translation thanks to Christoph Kuhn
	[FIXED] Direction "to here from" were flipped around
	[FIXED] Deleting domains in the API Key Settings works now
	[FIXED] Missing inclusion of the install tool class caused bug in TYPO3 versions < 4.1
1.3.0	[NEW] Domain management for API keys so the extension can be used on multidomain setups.
	[NEW] Add map controls to the FE User Map in the BE.
	[NEW] Directions are now built into the map, complete with written driving directions.
	[NEW] Add method that adds a marker based on just a table and uid. The address info is taken out of the TCA. See documentation for more details.
	[NEW] If there is only one marker on the map it's now possible to open the info bubble on load.
	[FIXED] potential bug that could cause funky errors due to clearing of cache when it wasn't necessary.
	[FIXED] "cannot redeclare $this" error message in PHP in a TYPO3 hook declaration.
	[FIXED] In some instances the geocoder would override manually changed coordinates.
	[FIXED] Make the map independent of the Prototype JS library. This fixes possible compatibility problems with other JS libraries and saves several hundred kb of files to be loaded.
	[FIXED] Don't include Google's API and other files more than once in the header if there is more than one map on a page.
	[FIXED] Some improvements to the geocoders.
1.2.0	[NEW] More than one map can now be put on one page. !!! To make this possible, the id of the containing div had to be made unique. That means if you used it in your CSS to style the map you will have to change it from the id "map" to the class "tx-wecmap-map". There is also a way to manually specify the id, see the TS reference for both plugins for more information. !!!
	[NEW] Initial center and zoom level can now be set via TS. See TS reference for more information.
	[FIX] Unescaped quotes causing the map to not show in certain cases
1.1.2	[FIX] Path error resulting in Prototype not being included properly in the BE
	[FIX] Extension can now be installed globally without errors
1.1.1	[NEW] Add status indicator image when saving a cache record
	[FIX] Improve rendering of record table with long coordinates
	[FIX] Bug that caused the current cache record's buttons to freeze
1.1.0	[NEW] Improved WEC Admin Geocode Admin including record filter, inline editing of records, sortable tables columns, speed improvements and overall design improvements, all using AJAX
	[NEW] API now supports tabs for marker info windows. See "Using the WEC Map API" for more information
	[NEW] API and plugins now support directions
	[FIX] Several bugs with geocode status in BE records not saving correctly
	[FIX] Deleted records are no longer geocoded and displayed
	[FIX] Geocode status now shows the right amount of significant digits
	[FIX] Variable name typo causing addresses in the FE User Map Plugin to not be mapped
1.0.1	[NEW] Add ability to set initial map type
	[NEW] Address fields are now read from the TCA
	[FIX] Some typos in localization
	[FIX] Replace short open tag with full open tag
	[FIX] Progress bar in Batch Geocoder now renders correctly
1.0.0	Initial release.