MapInfo2Leaflet

User Guide

TWIAV.NL

Version 0.30 – 12 October 2014

Table of Contents

1. Introduction

With **MapInfo2Leaflet** you can export a MapInfo table to a web map application using Leaflet - the output consists of an html file, a js file (containing a GeoJSON FeatureCollection) and css files for the styling.

Leaflet is a modern open-source JavaScript library for mobile-friendly interactive maps.

More information: http://leafletjs.com/

This version of **MapInfo2Leaflet** is compatible with the 'classic' 32 bit version of MapInfo Professional (version 9.5 or higher)

2. Export a MapInfo Table to a Web Map Application using Leaflet

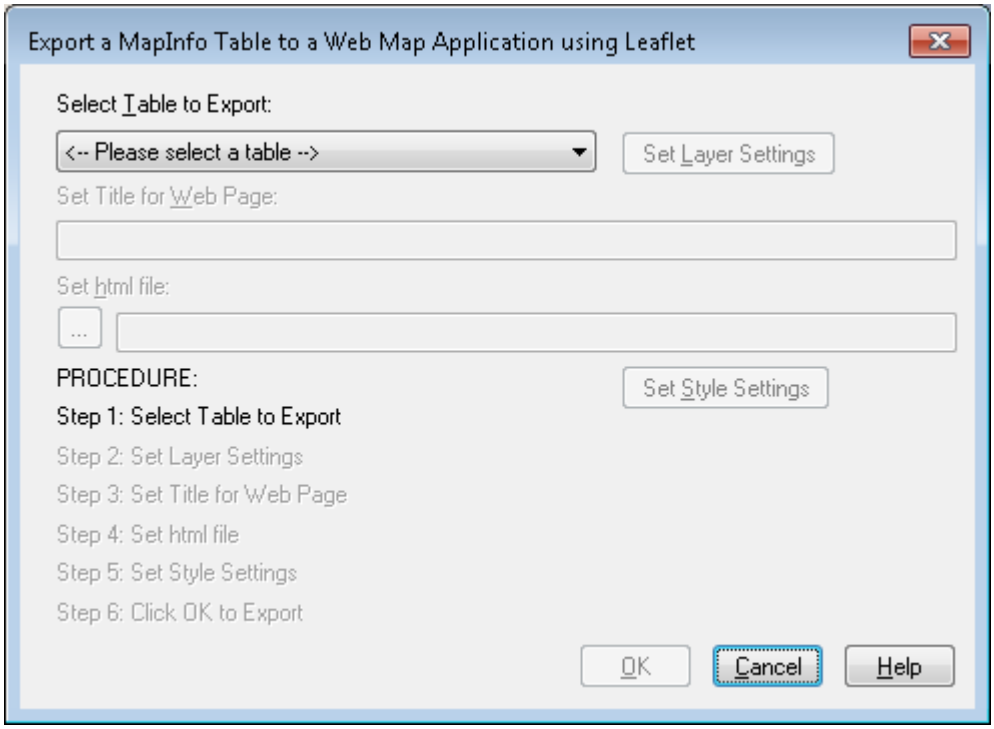
In MapInfo Professional open the table you want to export to a web map application.

The table to export should fall in one of the following categories:

- the table only contains **point data** (*point objects* and *multipoint objects*)
- the table only contains **line data** (*arc objects, line objects* and *polyline objects*)
- the table only contains **region data** (*ellipse objects, region objects, rectangle objects* and *rounded rectangle objects*)

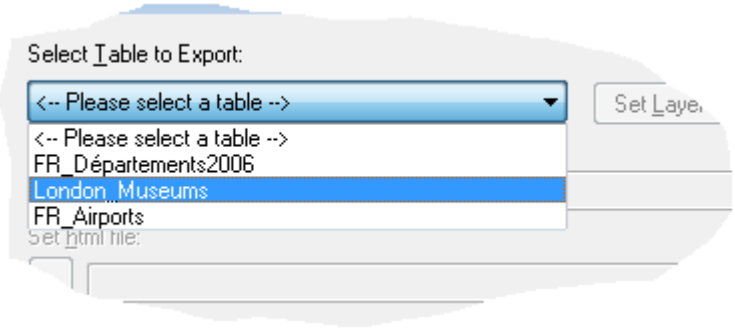
Tables containing a mixture of point, line and region data will not be processed by **MapInfo2Leaflet**.

Go to **MapInfo2Leaflet > MapInfo2Leaflet** or click the **MapInfo2Leaflet** button

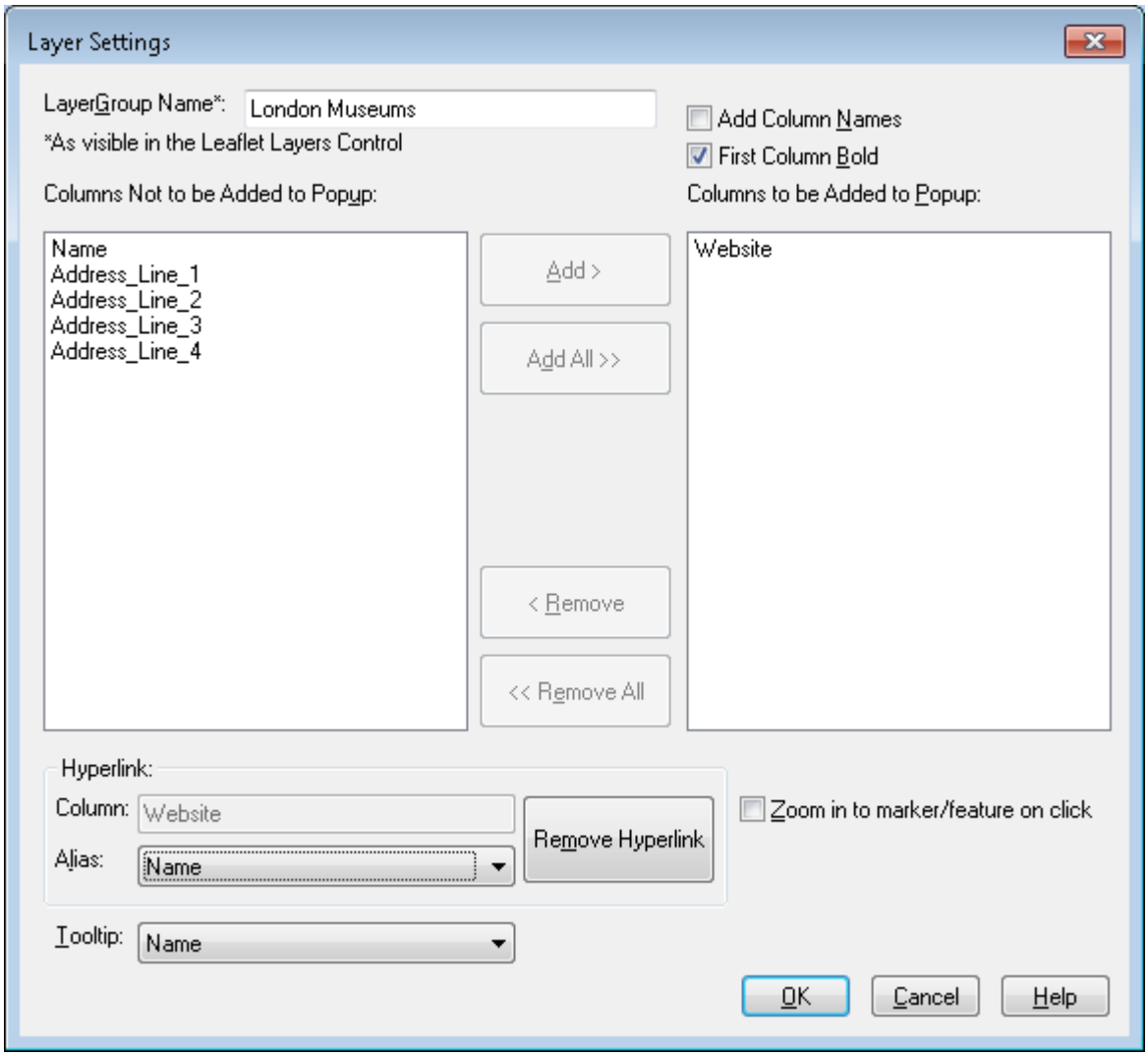


2.1. Step 1: Select Table to Export

Select your table from the available tables, e.g. ***London_Museums***. This will activate the **Set Layer Settings** button.



2.2. Step 2: Set Layer Settings



When you click the **Set Layer Settings** button, the **Layer Settings** dialog will appear.

In this dialog you can set the following options:

- **LayerGroup Name**

This name will be shown in the **Leaflet Layers Control** in the web map application, e.g. ***London Museums***.

- **Columns to be Added to Popup**

This defaults to all columns in the table, but you can **Remove** columns from the list. To reorder the columns in the popup: **Remove All**, and manually **Add** in the right order.

- **Add Column Names**

When you check this option, column names will also be shown in the popup

- **First Column Bold**

When you check this option, the first column value will be shown bold in the popup

- **Hyperlink**

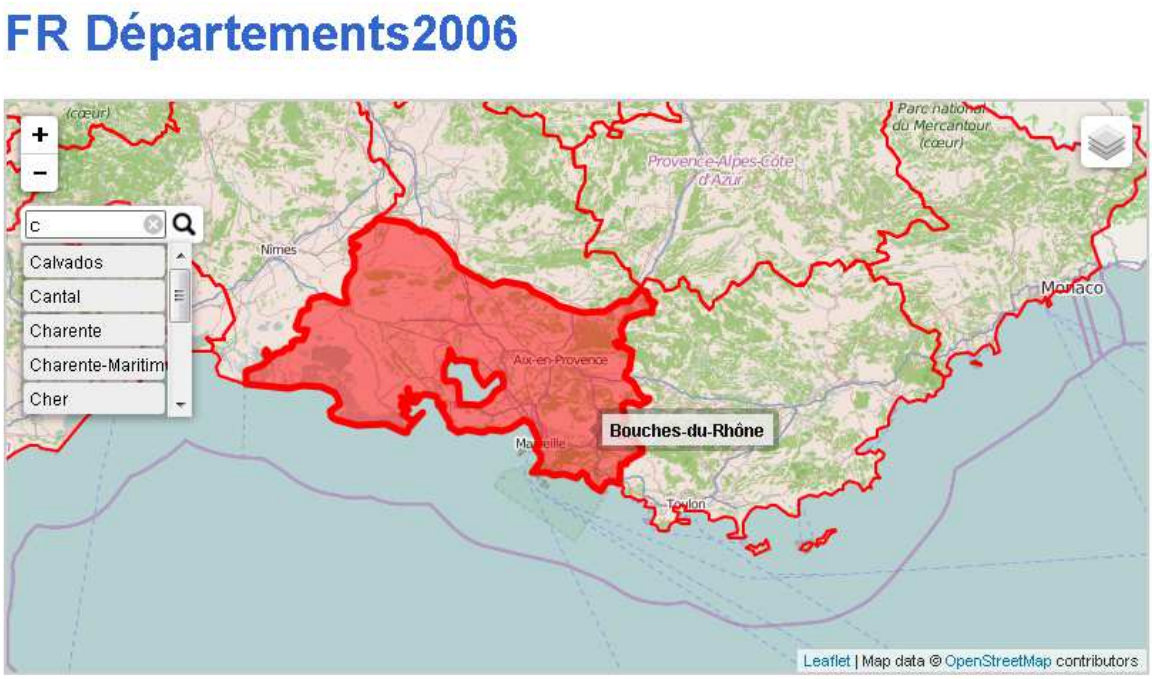
When processing the table **MapInfo2Leaflet** will automatically detect whether the table does contain a column with URLs, i.e. links to web pages. To be detected these URL values should start with “http”.

If a column with URLs is present a hyperlink will be created automatically.

You can set an **Alias** (the value of one of the other columns).

- **Tooltip**

This column will be shown in the tooltip/label, and it will also be used in the search control.



- **Zoom in to marker/feature on click**

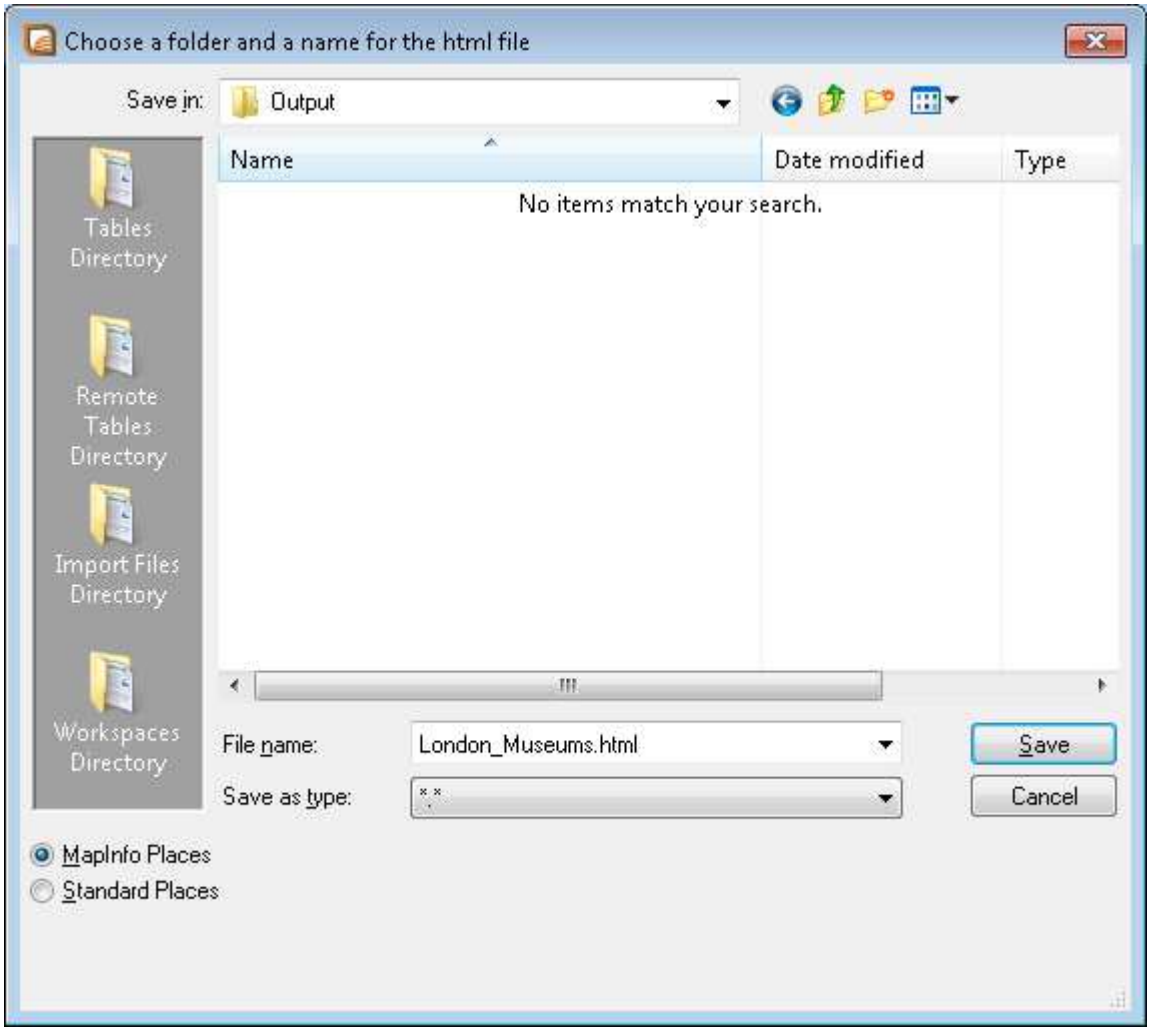
Be careful with this setting. Only use it on specific request by the end users. Why? What does this option do? Well, it makes sure that the map will be zoomed in on an object (marker or feature) when the user clicks this object. So, every time the user wants to open a tooltip by clicking an object the whole map will start moving.

Click **OK** to return to the **Export** dialog

2.3. Step 3: Set Title for Web Page

You can modify the title of the web page. The value you enter here will be used both in the TITLE element of the web page and in the header.

2.4. Step 4: Set html file



You can set the name and the output folder for the html file to be created.

It is **recommended** to choose an empty folder for this html file. This folder will not only be used to store the html file, but also all other components of the web map application. These other components will either be created in or copied to this folder. So after the export you will find multiple files and subfolders in the folder you specify here.

2.5. Step 5: Set Style Settings

In this release of **MapInfo2Leaflet** it is only possible to partly influence the style of point objects before the export of the table. All other style settings are set to default values. These values can be changed afterwards in the relevant html and css files.

- **Style for Region data**

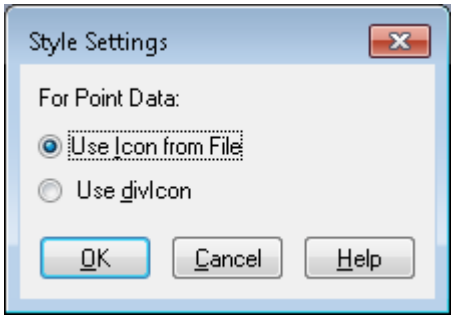
Transparent with red border

- **Style for Line data**

Red lines

- **Style for Point data**

When you click the **Set Style Settings** button, the **Style Settings** dialog will appear.



For point data you can choose between two styles:

- **Use Icon from File** 
- **Use divIcon** 

At the moment both options come with a default value, a blue arrow and a green dot with a red border respectively.

You can edit the html file after it has been created to modify the point style, i.e. use your own icon or change the colours of the circle.

To use another **Icon from File** modify or replace the variable specifying the iconUrl*:*

```
        var pngIconL1 = L.icon({
            iconUrl: 'blue-arrow.png',
            iconSize: [32, 37],
            iconAnchor: [16, 37],
            popupAnchor: [0, -28]
        });
```

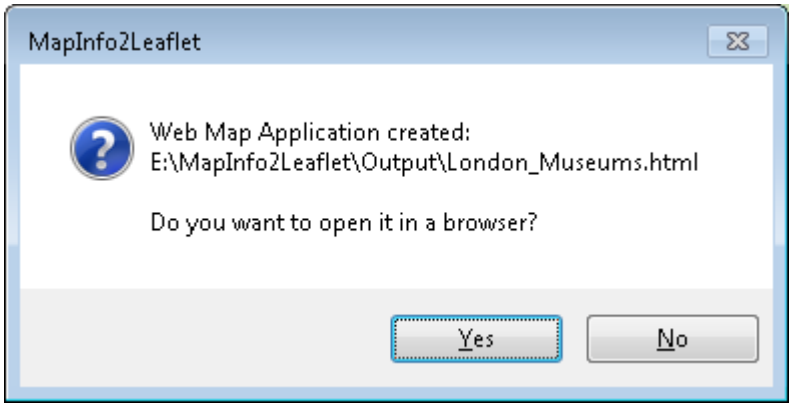
Please note: do not remove or rename the file *blue-arrow.png* in the application folder because the application needs this file to create the web map application. Only replace the *.png file in the destination folder.

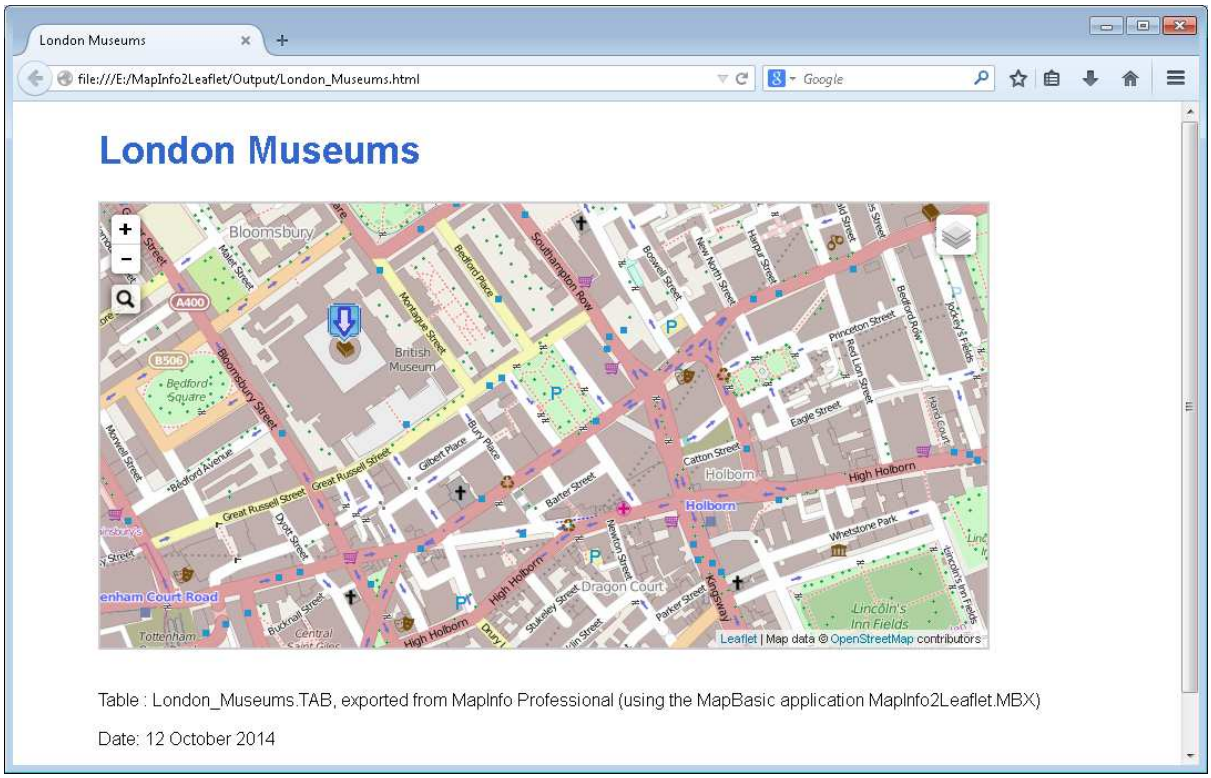
The appearance of the **divIcon** is specified in the mapinfo.css:

```
.miMarker {
    background-color: rgba(0,128,0,.9);
    border: 3px solid #FF0000;
    border-radius: 50%;
}
```

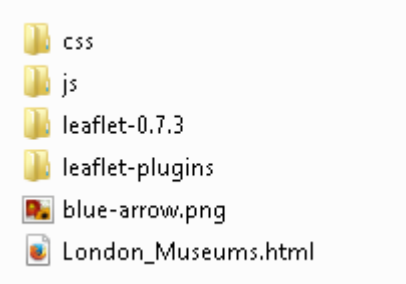
2.6. Step 6: Click OK to Export

When all settings are set you can click **OK** to create the web map application.





The output consists of multiple files and folders:



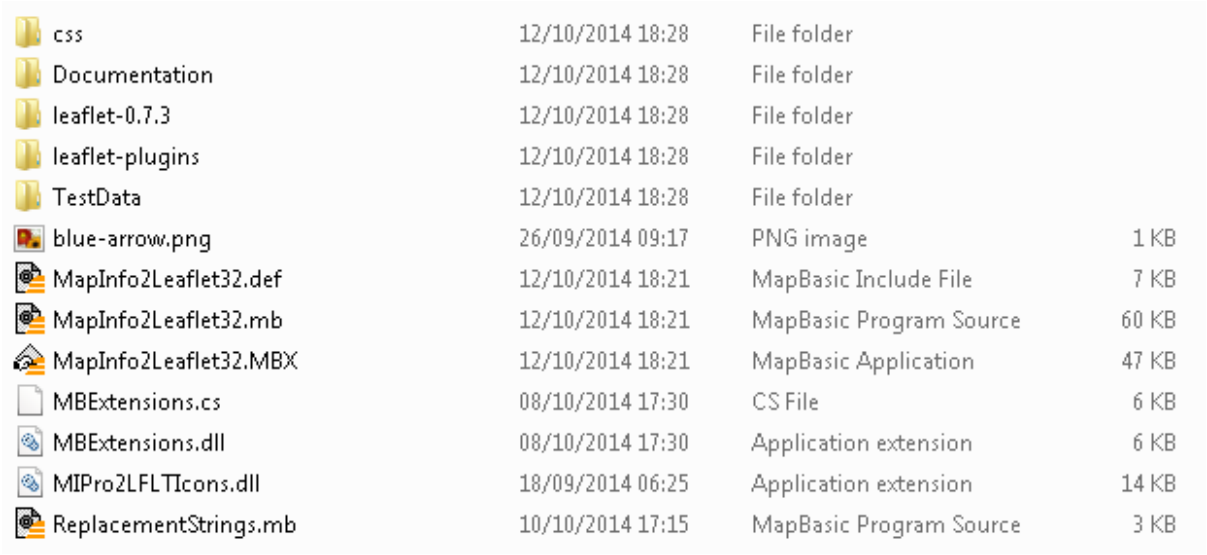
- An *.html file, e.g. **London_Museums.html**. This file contains the basic layout of the website – see chapter Sample Output for an example.
- A *.js file, e.g. **London_Museums.js** (in the subfolder js). This file contains the geometry, i.e. the point data (a GeoJSON FeatureCollection) – see chapter Sample Output for an example.
- All other files and folders are just copied to the output folder

3. Installation

3.1. MapInfo2Leaflet32Version030.zip

To install **MapInfo2Leaflet version 0.30** unzip **MapInfo2Leaflet32Version030.zip** in a folder of your choice.

The folder named **MapInfo2Leaflet32** does contain these files and folders:



(Source code included.)

3.2. Installation includes Leaflet library and plugins

The installation of **MapInfo2Leaflet** includes the Leaflet library in the folder `leaflet-0.7.3`.

The version included is Leaflet 0.7.3 (May 23, 2014).

Information on the latest version of Leaflet can be found here: http://leafletjs.com/download.html

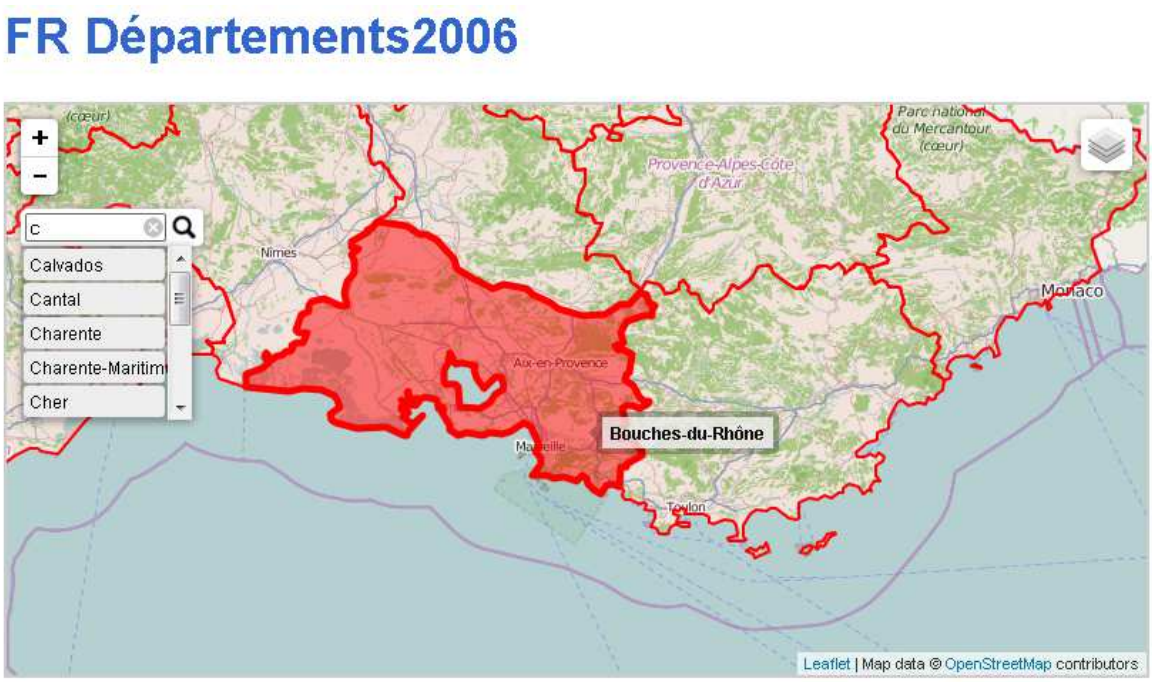
The installation of **MapInfo2Leaflet** also includes some Leaflet plugins in the folder `leaflet-plugins` .

These plugins are downloaded from the official Leaflet Plugins page:
http://leafletjs.com/plugins.html

The plugins currently included with **MapInfo2Leaflet** are:

- `Leaflet.label` by Jacob Toye. This adds text labels to map markers and vector layers.
- `leaflet-search` by Stefano Cudini. This adds a control for search Markers/Features location by custom property in GeoJSON.

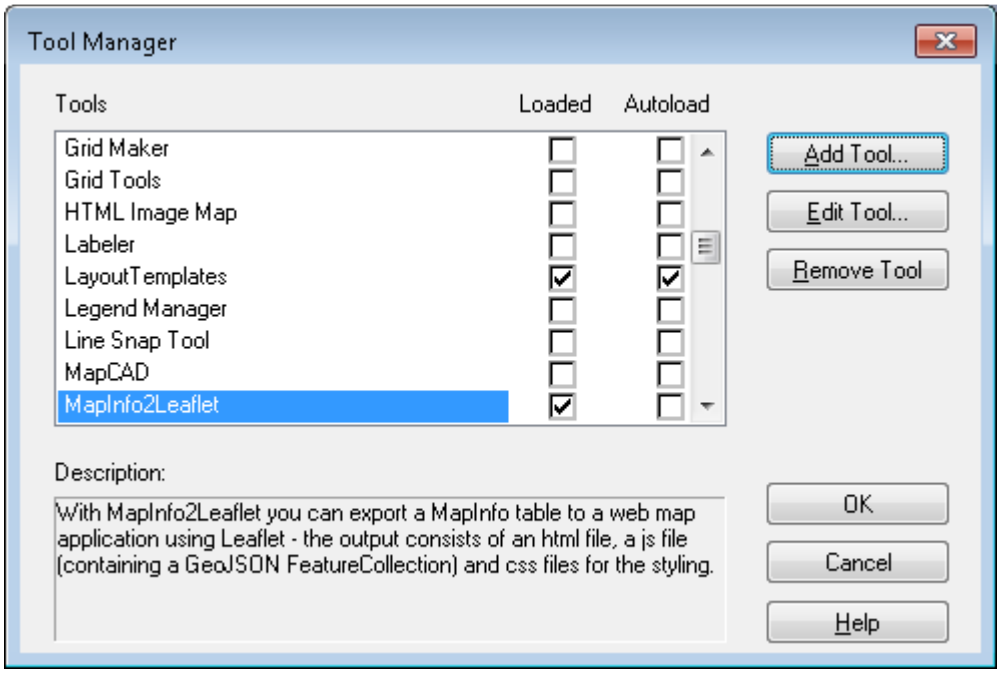
See screen dump below to see both plugins –and Leaflet itself of course - in action.



3.3. Run MapInfo2Leaflet: MapInfo2Leaflet32.MBX

To run **MapInfo2Leaflet** double-click `MapInfo2Leaflet32.MBX`

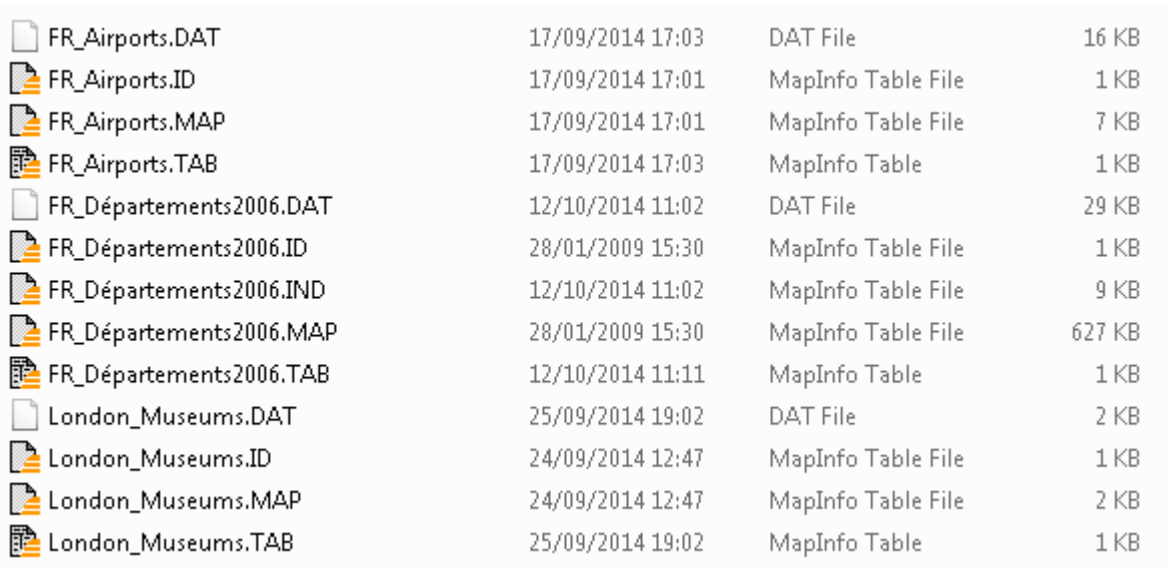
Optionally you can add the tool to the Tool Manager.



3.4. Test data

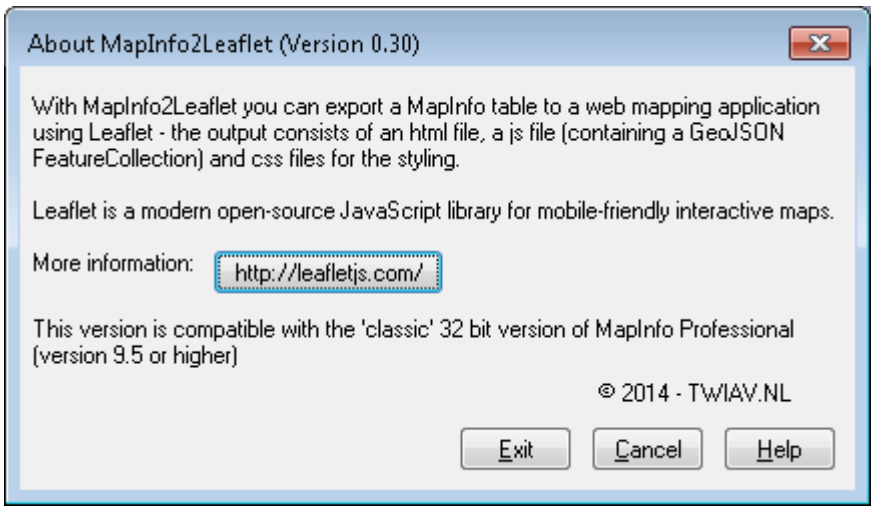
Some tables which have been used during development and testing are included:

London_Museums.TAB, **FR_Airports.TAB** and **FR_Départements2006.TAB**. These tables are in local projections, British National Grid and French Lambert-93, respectively.

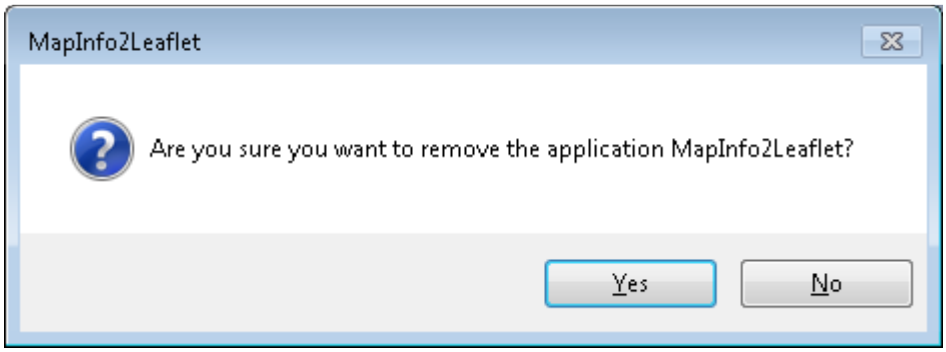


3.5. Unload the tool

To unload the tool from MapInfo Professional, go to **MapInfo2Leaflet > About MapInfo2Leaflet**



In the **About MapInfo2Leaflet** dialog, click the **Exit** button.



4. The Web Map Application – technical details

4.1. Leaflet 0.7.3

In the Web Map Application we use Leaflet (http://leafletjs.com/).

Leaflet is a modern open-source JavaScript library for mobile-friendly interactive maps.

The **MapInfo2Leaflet** installation contains local copies of Leaflet and some plugins. These are stored in the subfolders `leaflet-0.7.3` and `leaflet-plugins` respectively.

Both these folders are copied to the destination folder of your web map application.

4.2. Background map: OpenStreetMap

The background map is OpenStreetMap:

```
var osmAttrib='Map data © <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors',
osmUrl='http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png';
```

4.3. Input Projection: any projection

The input MapInfo Table can be in any projection. During the creation of the GeoJSON FeatureCollection **MapInfo2Leaflet** will extract the coordinates in WGS84 (EPSG:4326), independent of the projection of the input table.

4.4. Output Projection: EPSG:4326 (WGS84)

The projection used in the GeoJSON file is EPSG:4326.

The CRS object added contains the value

```
"crs":{"type":"EPSG","properties":{"code":"4326"}}
```

Please note: as mentioned in the previous paragraph the input MapInfo Table can be in any projection. **MapInfo2Leaflet** will convert the coordinates to WGS84 (EPSG:4326) during the export.

4.5. Styling

Some basic styling of the web page is provided by a set of cascading style sheet (css) files: **page.css** – with some general style settings - and **mapinfo.css** – containing map specific settings. You can modify (or replace) this style sheet to suite your needs.

5. Sample Output

5.1. html file

E:\MapInfo2Leaflet32\Output\London_Museums.html

```
<!DOCTYPE html>
<html>
<head>
    <title>London Museums</title>
    <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="leaflet-0.7.3/leaflet.css" />
    <link rel="stylesheet" href="leaflet-plugins/leaflet.label.css" />
    <link rel="stylesheet" href="leaflet-plugins/leaflet-search.min.css" />
    <link rel="stylesheet" href="css/page.css" />
    <link rel="stylesheet" href="css/mapinfo.css" />
</head>
<body>
    <header>
        <h1>London Museums</h1>
    </header>
    <div id="main">
        <div id="map"></div>
        <br>
        <p>Table : London_Museums.TAB, exported from MapInfo Professional (using the MapBasic application
MapInfo2Leaflet.MBX)</p>
        <p>Date: 12 October 2014</p>
    </div>
    <script src="js/London_Museums.js"></script>
    <script src="leaflet-0.7.3/leaflet.js"></script>
    <script src="leaflet-plugins/leaflet.label.js"></script>
    <script src="leaflet-plugins/leaflet-search.min.js"></script>
    <script>
        // The layer london_museumsLayer is referred to as 'L1', i.e. all functions and variables specific to this
layer have the suffix L1
        // These are the bounds of the london_museums dataset, plus a margin
        var southWest = L.latLng(51.5059, -0.131729),
```

```
            northEast = L.latLng(51.5212, -0.0946247),
            bounds = L.latLngBounds(southWest, northEast);

        var map = L.map('map').fitBounds([[51.5171, -0.13307], [51.5221, -0.123223]]).setMaxBounds(bounds);

        // Limit the minimum zoom level to the bounds of the dataset to prevent users from leaving this area on the
map
        var mz = map.getBoundsZoom(bounds, false);

        var osmAttrib='Map data © <a href="http://openstreetmap.org">OpenStreetMap</a> contributors',
            osmUrl='http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png';

        var osm = L.tileLayer(osmUrl, {attribution: osmAttrib, minZoom: mz}).addTo(map);

        var pngIconL1 = L.icon({
            iconUrl: 'blue-arrow.png',
            iconSize: [32, 37],
            iconAnchor: [16, 37],
            popupAnchor: [0, -28]
        });

        function onEachFeatureL1(feature, layer) {
            if (feature.properties && feature.properties.popupContent) {
                layer.bindPopup(feature.properties.popupContent);
            }
            if (feature.properties && feature.properties.name) {
                layer.bindLabel(feature.properties.name);
            }
        }

        var london_museumsLayer = L.geoJson(london_museums, {
            pointToLayer: function (feature, latlng) {
                return L.marker(latlng, {icon: pngIconL1});
            },
            onEachFeature: onEachFeatureL1
        }).addTo(map);

        var baseLayers = {
            "OpenStreetMap": osm
```

```
        };

        var overlays = {
            "London Museums": london_museumsLayer
        };

        L.control.layers(baseLayers, overlays).addTo(map);

        var searchControl = new L.Control.Search({layer: london_museumsLayer, propertyName: 'name',
circleLocation:false});

        searchControl.on('search_locationfound', function(e) {
            e.layer.setStyle({fillColor: '#0000FF', color: '#0000FF'});
            if(e.layer._popup)
                e.layer.openPopup();
        }).on('search_collapsed', function(e) {
            featuresLayer.eachLayer(function(layer) { //restore feature color
                featuresLayer.resetStyle(layer);
            });
        });

        map.addControl(searchControl); //inizialize search control

    </script>
</body>
</html>
```

5.2. Javascript file

E:\MapInfo2Leaflet\Output\js\London_Museums.js

```
//Table: London_Museums.TAB (exported from MapInfo Professional using MapInfo2Leaflet.MBX)
//Date: 12 October 2014
var london_museums = {"type":"FeatureCollection","features": [
{"type":"Feature","properties":{"name":"British Museum","address_line_1":"Great Russell
Street","address_line_2":"London","address_line_3":"WC1B
3DG","website":"http://www.britishmuseum.org/","popupContent":"<b>British Museum<\/b><br>Great Russell
Street<br>London<br>WC1B 3DG<br><a
href=\"http://www.britishmuseum.org/\">http://www.britishmuseum.org/<\/a>"},"geometry":{"type":"Point","coordinates"
:[-0.126981, 51.5195]}},
{"type":"Feature","properties":{"name":"Tate Modern","address_line_1":"Bankside","address_line_2":"London SE1
9TG","address_line_3":"United Kingdom","website":"http://www.tate.org.uk/visit/tate-modern","popupContent":"<b>Tate
Modern<\/b><br>Bankside<br>London SE1 9TG<br>United Kingdom<br><a href=\"http://www.tate.org.uk/visit/tate-
modern\">http://www.tate.org.uk/visit/tate-modern<\/a>"},"geometry":{"type":"Point","coordinates":[-0.099381,
51.5076]}},
{"type":"Feature","properties":{"name":"London Transport Museum","address_line_1":"Covent Garden
Piazza","address_line_2":"London WC2E 7BB","website":"Unknown","popupContent":"<b>London Transport
Museum<\/b><br>Covent Garden Piazza<br>London WC2E 7BB<br>Unknown"},"geometry":{"type":"Point","coordinates":[-
0.121304, 51.5119]}}
],"crs":{"type":"EPSG","properties":{"code":"4326"}}};
```