

Geographische Karten einbinden mit Drupal und Leaflet - Alternative zu Google Maps

Kürzlich bekamen wir eine Anfrage für die Umsetzung einer Karte mit Leaflet und Openstreetmaps.

Da es ja diverse Gründe gibt, sich nicht vom Lizenz-Modell von Google abhängig zu machen, haben wir die Anfrage zum Anlass genommen, die Darstellung der Lieferanten auf Deutschland-Karten auf der Webseite über [Gartenstauden](#) und [Baumschul-Pflanzen](#) auf Leaflet und Openstreetmap umzustellen.

[Leaflet](#) ist natürlich nicht das einzige geeignete Modul.

Für die Leaflet-Open-Source-Library in JavaScript spricht, daß sie sehr schnell und für mobile Endgeräte geeignet und somit zukunftsfähig ist.

Außerdem gibt es eine gute Dokumentation und eine lebendige Community, die es erlaubt, Sonderwünsche zu programmieren.

Die reine Darstellung der Lieferanten auf der Karte wäre ganz einfach mit Drupal-Boardmitteln umzusetzen.

Nun ist die Übersichtlichkeit auf der Karte unbefriedigend, wenn viele Einträge (Marker) in einem engen lokalen Bereich auftauchen.

Dies versucht man durch verschiedene Cluster-Methoden zu umgehen.

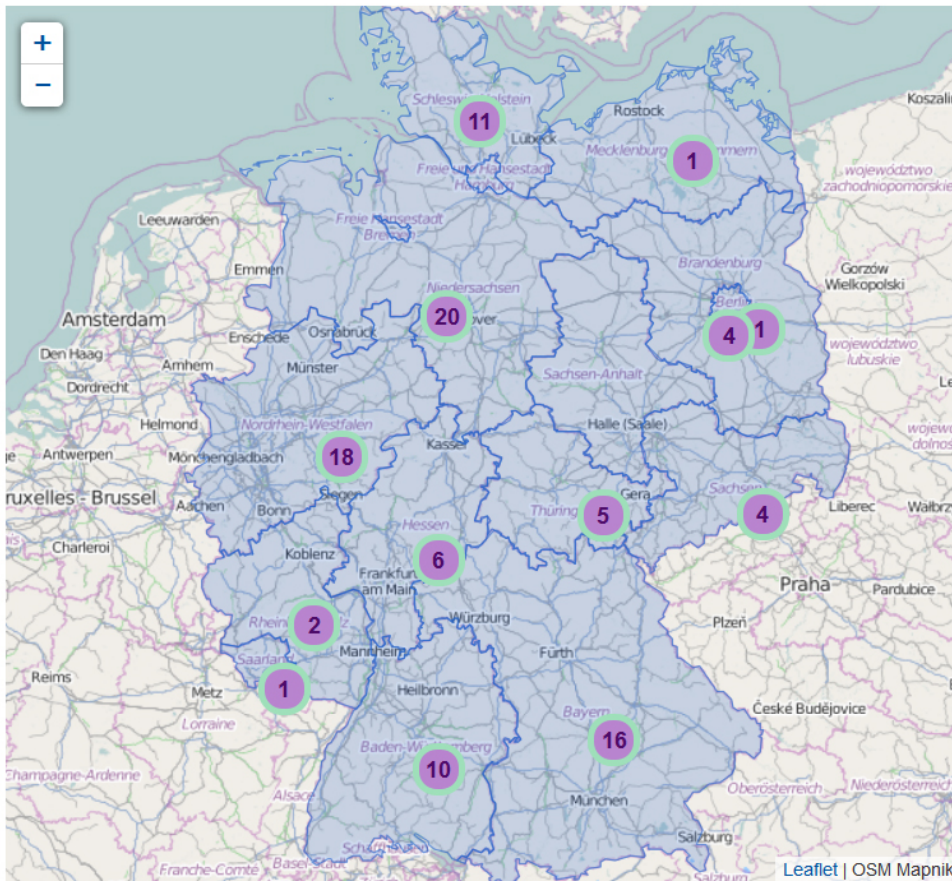
Clustering bedeutet, daß Marker, die in einer bestimmten Entfernung liegen, bei einem bestimmten Zoomfaktor zusammen gefasst werden und erst beim Zoomen in die Karte werden die einzelnen Marker dargestellt.

In unserem Fall soll das Clustering nicht von der Entfernung der Marker abhängen, sondern es sollen beim Aufruf der Karte die Grenzen der Bundesländer gezeigt werden und ein Kreis mit der Anzahl der Lieferanten für dieses Bundesland.

Beim MouseOver soll das Bundesland farbig hervorgehoben werden.

Bei einem Click auf das Bundesland wird dieses vergrößert dargestellt und die Betriebe als einzelne Marker dargestellt.

Alle Lieferanten auf der Karte



Beispiele für Clustering sind nur für das Standard-Cluster-Verfahren abhängig von der Entfernung der Marker zu finden.

Das funktioniert mit dem Cluster-Modul für Leaflet unter Drupal.

Für das Clustering pro Bundesland und die farbige Darstellung kommt deshalb individuelles JavaScript (jQuery) und das GeoDB-Modul von Leaflet zum Einsatz.

Im Folgenden beschreibe ich die Schritte, die für die Umsetzung notwendig sind:

1. Module installieren:

- [leaflet](#)
- [libraries](#)
- [leaflet_geojson](#)
- [leaflet_widget](#)
- [addressfield](#)

- [geocoder](#) (bitte nicht mit geocode für Drupal 6 verwechseln)
- [geofield](#)
- [geoPHP](#)

Diese Leaflet- Module werden aktiviert:

☒	Leaflet	7.x-1.1
☒	Leaflet GeoJSON	7.x-1.0
☐	Leaflet GeoJSON Bean	7.x-1.0
☒	Leaflet views	7.x-1.1
☒	Leaflet Widget for Geofield	7.x-2.0-beta1

2. Library für Leaflet installieren

Die JavaScript Dateien von Leaflet werden vom [Download-Verzeichnis der Leaflet-Homepage](#) geholt und in das Verzeichnis sites/all/libraries geschoben.

3. Inhaltstyp Pflanzen-Lieferanten wird um folgende Felder ergänzt:

Hiermit werden mehrere Felder für die Adress-Informationen angelegt.
Das Feld ersetzt also die Informationen, die bisher in Location gespeichert sind.

+ Neues Feld hinzufügen			
Adresse	field_adresse [Bearbeiten]	Postal address	Dynamic address form
Beschriftung		Der zu speichernde Datentyp.	Formularelement zum Bearbeiten der Daten.

Nun benötigen wir noch ein Feld, das aufnimmt, welchem Bundesland unsere Adresse angehört.

Neues Feld hinzufügen

Bundesland field_bundesland [Bearbeiten] Textfeld

Beschriftung Der zu speichernde Datentyp. Formularelement zum Bearbeiten der Daten.

Die Felder werden alle als Pflichtfeld angelegt.

4. Beim Inhaltstyp Ansicht der neuen Felder ändern:

Geocoder:

Geocoder Formateinstellungen: **Leaflet**

Leaflet Map *

OSM Mapnik

Map Height

400 px

POPUP SETTINGS

ZOOM SETTINGS

These settings will override the zoom settings defined in the map definition. are 'zoomed out', high values are 'zoomed in'.

Initial zoom level

Use map defined setting

The starting zoom level when this map is rendered. Restricted by min and max zoom level.

Minimum zoom level

1

The minimum zoom level allowed. (How far away can you view from?)

Maximum zoom level

Use map defined setting

The maximum zoom level allowed. (How close in can you get?)

POINT ICON

These settings will overwrite the icon settings defined in the map definition.

Icon Source

☒ Icon File

☐ Field (html DivIcon)

Icon URL

/sites/all/themes/gartenstauden/images/leafletmarker.png

Hier kann man einfach ein eigenes Icon für den Marker hinterlegen, wenn man nicht den Drupal-Wassertropfen verwenden möchte.

Da wir hier die Einstellungen für die Karte einrichten, die in der Detailansicht gezeigt wird, benötigen wir kein PopUp auf dem Marker, da alle Infos bereits auf der Seite stehen.

5. Daten aus bisherigem Adressfeld in die neuen Felder übertragen

Um die Ansicht der Karte zu testen, übertragen wir bei einem Datensatz händisch die Daten.

Nun sehen wir unsere Karte noch in einem zu starken Zoom. Das ändern wir nachher im JS.

Da sich niemand findet, der die ganzen Daten händisch verpflegen möchte und das Modul Feeds bereits eingerichtet ist, bauen wir uns eine View, die die bisherigen Adressdaten (Nid, Straße, PLZ, Ort) in einer CSV-Datei zurück gibt.

Aus einer geeigneten Open-Source-Quelle besorgen wir uns eine Tabelle mit Bundesland->PLZ-Zuordnung.

Die beiden CSV Dateien kopieren wir in zwei Tabellenblätter eines Open Office-Calc (bzw. Excel) Dokuments und schreiben eine Formel, die zu jedem Nid über die PLZ das Bundesland ermittelt.

Davon erzeugen wir eine CSV-Datei und erstellen ein passendes Feeds-Mapping zum Import der Daten.

Nun sind unsere bisherigen Datensätze der Lieferanten mit den Daten im Adressfeld und im Bundesland befüllt.

Dieses Verfahren wird bei anderen Installationen überflüssig sein oder von dem beschriebenen Vorgehen abweichen.

6. View bauen für die Karte

Wir legen eine einfache Inhalts-View an mit Selektion auf den Inhaltstyp Pflanzen-Lieferanten und wählen Leaflet Map als Format.

Die weiteren Settings machen wir erst, wenn wir Felder angelegt haben, woran uns auch der Hinweis erinnert, wir sollten mindestens ein Feld mit Geo-Koordinaten bereit stellen.

Die Grundkonfiguration des Felds geocoder sieht so aus:

Configure field: Inhalt: Geocoder

Appears in: node:pflanzen_lieferanten.

☐ Create a label

Enable to create a label for this field.

☐ Exclude from display

Enable to load this field as hidden. Often used to group fields, or to use as token in another field.

Formatter

Well Known Text (WKT)



Data options *

Use full geometry



► **STYLE SETTINGS**

► **NO RESULTS BEHAVIOR**

► **REWRITE RESULTS**

► **MEHR**

zusätzlich legen wir noch Felder für Titel (als Link), Strasse, PLZ und Ort an.
In einem zusätzlichen Feld überschreiben wir den Wert mit einem String aus diesen Werten.

Der String sieht dann so aus:

```
<b>[title]</b><br>
[field_lieferanten_strasse]<br>
[field_lieferanten_plz] [field_lieferanten_ort]<br>
[field_bundesland]<br>
```

Das Feld heißt title_1.

Nun gehen wir in die Settings vom Format Leaflet Map und machen dort die

fehlenden Einstellungen:

Data Source *
Inhalt: Geocoder ▼
Which field contains geodata?

Title Field
Inhalt: Titel ▼
Choose the field which will appear as a title on tooltips.

Description Content
Inhalt: Titel ▼
Choose the field or rendering method which will appear as a description on tooltips or popups.

Map *
OSM Mapnik ▼

Map height
550 px

Das Titel Field bezieht sich auf das erste Feld Titel, das wir angelegt haben. In Description steht der zweite Titel, der mit dem zusammengebauten String überschrieben wird. Bei der Map wählen wir wieder Openstreetmap und als Höhe 550px.

Diese Höhe ist gewählt, weil dann Deutschland bei einem Zoomwert von 6 genau hinein passt.

Unter Point Icons geben wir den Pfad auf ein eigenes Marker-icon an:

▼ **POINT ICON**
These settings will overwrite the icon settings defined in the map definition.

Icon Source
☒ Icon File
☐ Field (html DivIcon)

Icon URL
/sites/all/themes/gartenstauden/images/leafletmarker.png
Can be an absolute or relative URL.

Icon Shadow URL

Can be an absolute or relative URL.

▼ **ICON SIZE**
It is usually preferable to leave this blank and use the image style system to size the icons. Sizes are in pixels.

Breite
12

Höhe
20

Nun geben wir noch den Pfad an, unter dem wir die Seite mit der Karte sehen möchten und schauen uns das Ganze an.

Die Karte wird in der korrekten Größe auf Deutschland optimiert angezeigt und ebenso die Marker mit unserem eigenen Icon.

Da sehr viele Icons auf relativ kleinem Raum angezeigt werden, ist die Karte noch unübersichtlich.

Deshalb geht es jetzt an die Feinarbeit mit dem Ziel, zuerst nur Umriss von den Bundesländern zu zeigen.

Jedes Bundesland soll einen Marker enthalten, der die Anzahl der Betriebe in diesem Bundesland wieder gibt (Clustering).

Für das Cluster-Verfahren bringt Leaflet sogar ein eigenes Modul mit (Leaflet Markercluster) .

Dieses Modul bietet Clustering nach Distanz. D.h. Man kann die Entfernung angeben, unter der benachbarte Marker geclustert angezeigt werden.

Beim Zoomen in die Karte werden die Marker dann jeweils ungeclustert gezeigt.

Wie oben schon geschrieben ist dieses Modul für unser Vorhaben nicht geeignet.

Deshalb werden weitere Feinheiten per JavaScript (Jquery) umgesetzt.

Feinarbeiten per JavaScript (Jquery)

Dafür bietet Leaflet eine ausführliche Doku zur Library. Noch hilfreicher sind allerdings die versch. Beispiele im Netz, die wir nach und nach zusammen gesammelt haben.

Folgende Aufgaben wurden per JS gelöst:

- Definiere Koordinaten + Zoom, auf die auch bei Klick auf die Karte ausserhalb DE zurueckgesprungen wird

```
map.setView(new L.LatLng(initialLat, initialLng), initialZoom);
```

- Gruppen von Layern bilden, die später unterschiedliche behandelt werden können
z.B. alle Marker, alle Layer für Bundesländer, alle Cluster-Layer
- Default Styles und Styles für MouseOver initialisieren:

```
var defaultStyle = { color: '#2262CC', weight: 1, opacity: 0.6,
```

- Eine Schleife machen über alle GeoJSON **Features** mit Bundesländern in Deutschland.

Diese Datei wurde aus einer Open-Source-Quelle bezogen und lokal abgelegt.

```
jQuery.getJSON(url, function (json) { L.geoJson(json, { onEachFeature  
this.setStyle(MouseOverStyle); }); }).addTo(map); }
```

Mit `addTo(map)` wird jeder erzeugte Layer an die vorhandene Karte hinzugefügt.

Ebenfalls für jedes Bundesland wird die Anzahl der Marker in diesem Bundesland ermittelt und mit?

```
myTextLabel = L.marker(textLatLng, { title: countmarker+' '+betrieb+' '});
```

angehängt.

Bei einem Klick auf das jeweilige Bundesland wird (u.a.) dieses Text-Label mit der Anzahl der Betriebe auf Unsichtbar gesetzt und dafür die Marker für jeden einzelnen Betrieb angezeigt.

Für all diese Funktionen werden die oben vorinitialisierten Layer-Gruppen verwendet und in einer Schleife durchlaufen.

- Map auf Ursprungs-Ansicht zurück setzen, sobald jemand auf einen Punkt außerhalb von Deutschland klickt

```
map.on({  
  click: function() {  
    //Map wieder mit Ausgangs Koord. und Zoom positionieren  
    map.setView(new L.LatLng(initialLat, initialLng), initialZoom);  
    //Marker alle verschwinden lassen  
    for( i in markerlayer._layers){  
      markerlayer._layers[i]._icon.style.display = 'none';  
    }  
    //Alle Clustermarker wieder sichtbar stellen  
    for( i in clusterlayer._layers){  
      clusterlayer._layers[i]._icon.style.display = 'Block';  
    }  
  }  
});
```

```
    }  
    //Polygone in Defaultstyle versetzen  
    for( i in geolayer._layers){  
        geolayer._layers[i].setStyle(defaultStyle);  
    }  
}  
});
```

Bitte schaut Euch das JavaScript einfach im laufenden Betrieb an, um die genaue Syntax zu finden.

Hier die verschiedenen Seiten, auf denen Leaflet-Karten eingebunden sind:

[Anzeige der Lieferanten auf der Deutschlandkarte](#)

[Übersicht über die Lieferanten mit Detail-Karte](#)