Documentation de config.ini

config.ini permet de configurer la génération de la carte pour le nœud rudi.
Il permet la creation d'une page html à partir d'un fichier geojson.

Par défaut avec un fichier config.ini vide, tous les éléments géométriques lus (Point , LineString , Polygon , MultiPoint , MultiLineString , and MultiPolygon) sont tous mis sur la carte.

Structure du fichier config.ini :

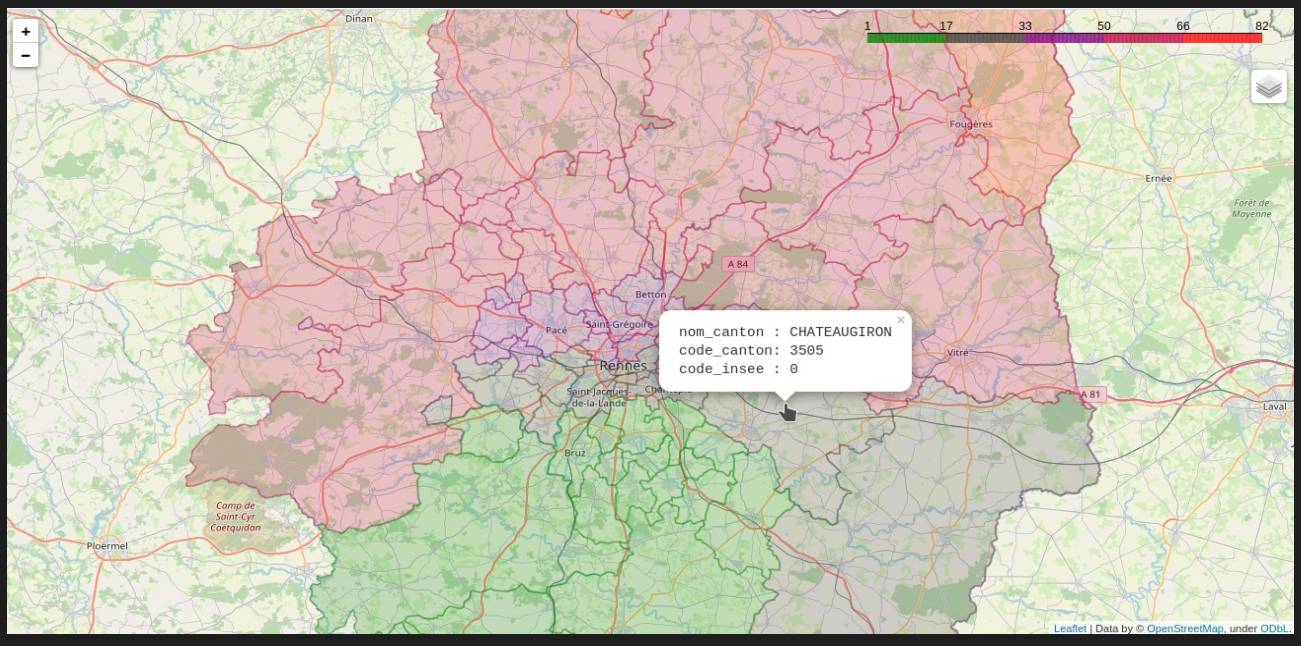
```
[METADATA]

# nom de la carte généré
title = generated_map

# nom du fichier geojson
geojson = Polygon/cantons-dille-et-vilaine-version-rennes-metropole.geojson

[MAP]

# afficher les points
markers = True

# assembler les points en clusters
clusterize_markers = True

# afficher la heatmap des points
heatmap = True

# valeur utilisé pour la heatmap (si non indiqué le poids de chaque point est le même)
heatmap_weight_field = nb_bacs_tot

# afficher les lignes
lines = True

# Si précisé, limite le nombre d'éléments affichés,
# les éléments les plus proches du centre de Rennes sont priorisés.
max_number_of_features_displayed = 500

# afficher les polygones
polygons = True

# rendre les polygones invisible (utile pour voir les informations de la zone en cliquant sur les zones)
#polygons_invisible = True

# afficher un choropleth: dans ce cas, polygons_invisible est True par dédault.
choropleth = True

# nom de la layer du choropleth affiché dans l'option en haut à droite
choropleth_name = nom du choropleth
```

```
# nom affiché dans la légende le la color map en haut à droite
choropleth_value_name = nom valeur dans la légende

# les polygones sont regroupé par groupes ayant la même valeur pour cette propriété
choropleth_key = nom_canton

# nom de la propriété dont la valeur est utilisé pour déterminer la couleur de la zone (une quatité en général)
choropleth_value = code_num

# liste python de string de couleurs. C'est la color map du gradient de couleurs
choropleth_colors = ['green','purple','red']

# Si précisé la couleur a un nombre maximum de couleurs différentes possible à la place de faire un gradient avec des transition de couleurs continues
choropleth_color_steps = 5

[FIELDS]

# Si à True, tous les champs sont affichés
#all = True

# 1ier champ
champ0 = nom_canton

# 2nd champ
champ1 = code_canton

#... n'importe quel autre nombre de champs

# liste des champs obligatoires pour qu'une éléments géométrique soit affichée (les éléments n'ayant pas au moins un des champs sont ignorés).
required_fields = ['nom_canton']
```

Exemples :

Exemple pour afficher un choropleth

```
[METADATA]

title = generated_map

geojson = cantons-dille-et-vilaine-version-rennes-metropole.geojson

[MAP]

markers = False
clusterize_markers = False

heatmap = False

lines = False

polygons = True

choropleth = True
choropleth_name = nom du choropleth
choropleth_value_name = nom valeur dans la légende
choropleth_key = nom_canton
choropleth_value = objectid ; pas pertinent, juste à titre d'exemple
choropleth_colors = ['green','purple','red']
choropleth_color_steps = 5

[FIELDS]

champ0 = nom_canton
champ100 = code_insee
champ87 = code_canton
required_fields = ['nom_canton']
```



Exemple heatmap and cluster:

[METADATA]

title = generated_map

geojson = points-presentation-bacs-roulants.geojson

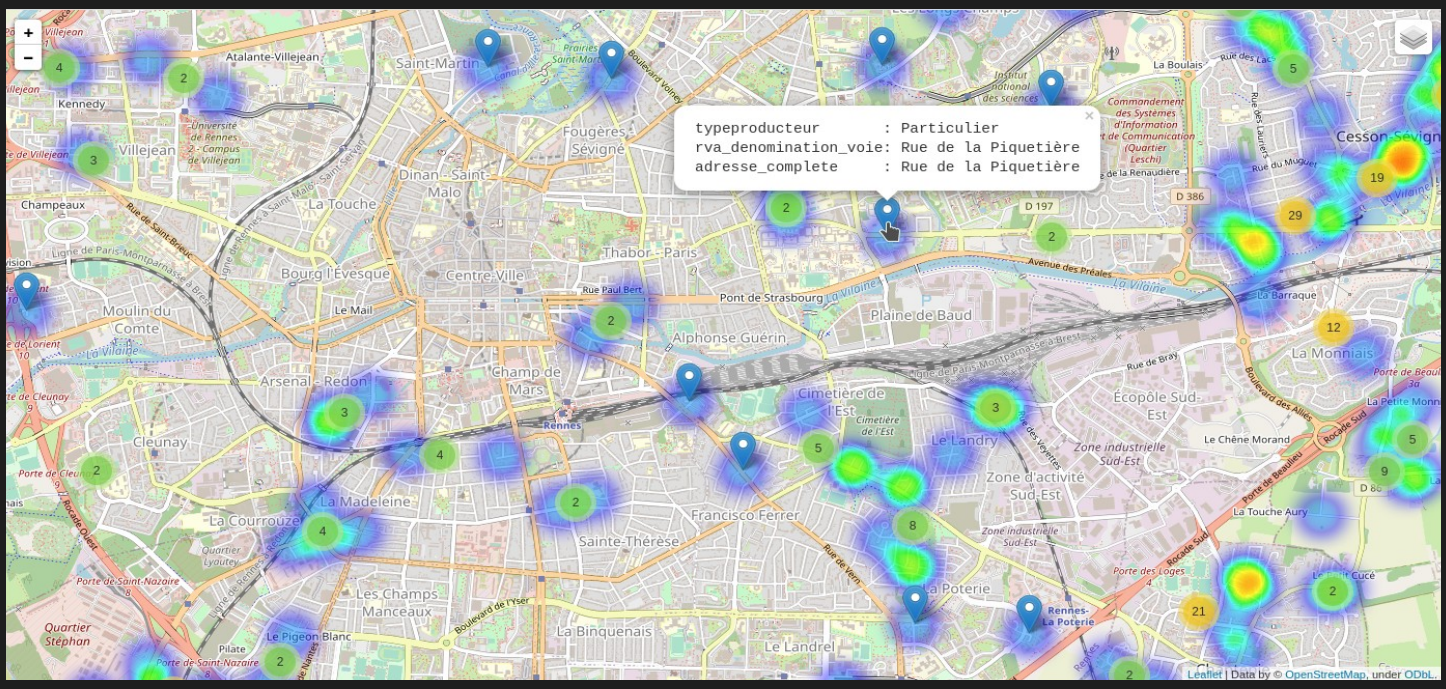
[MAP]

markers = True
clusterize_markers = True

heatmap = True
heatmap_weight_field = nb_bacs_tot

[FIELDS]

champ0 = typeproducteur
champ2 = rva_denomination_voie
champ5 = adresse_complete



Exemple Lines avec un nombre de éléments limité à 500 :

[METADATA]

title = generated_map

geojson = amenagements-velo-et-zones-de-circulation-apaisee.geojson

[MAP]

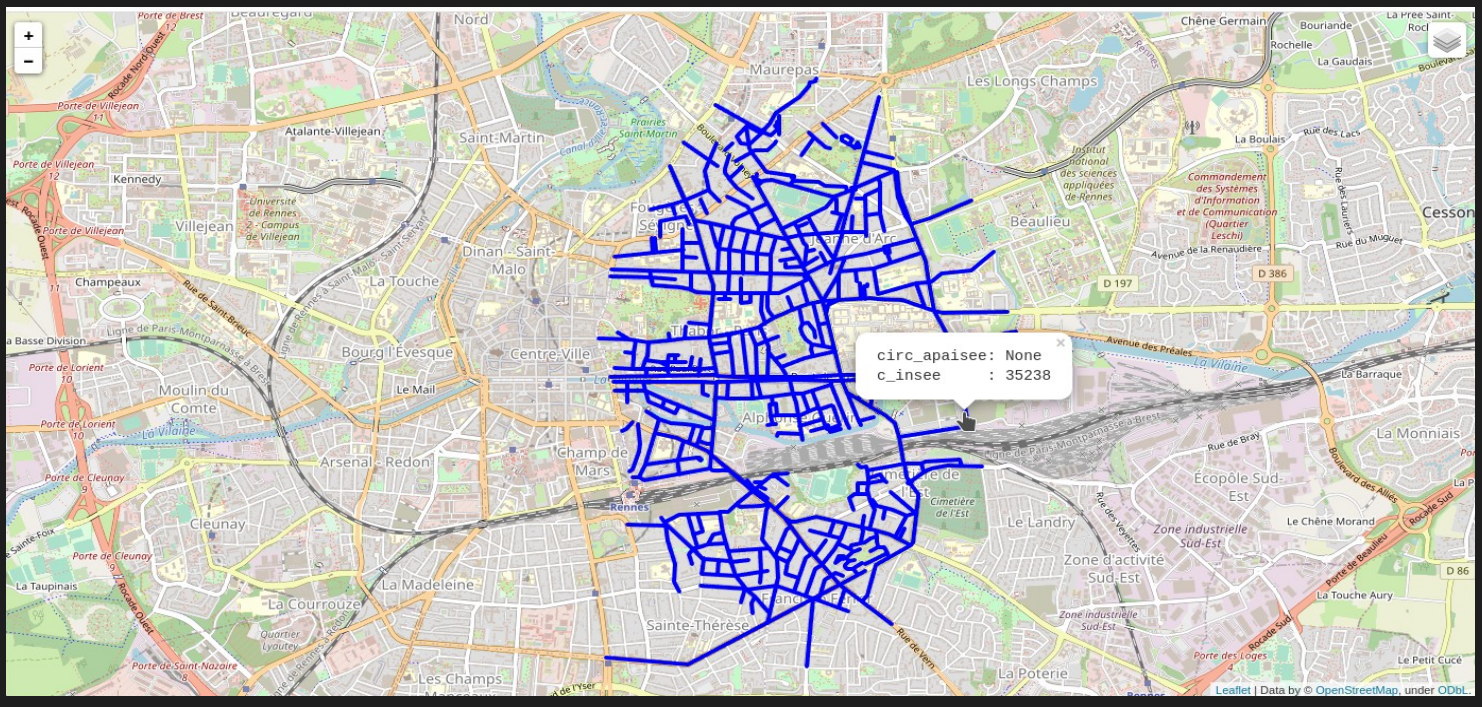
markers = False

lines = True
max_number_of_features_displayed = 500

polygons = False

[FIELDS]

champ0 = circ_apaisee
champ1 = c_insee



Exemple avec des polygones :

[METADATA]

title = generated_map

geojson = aires-de-jeux-des-espaces-verts-rennes.geojson

[MAP]

polygons = True

[FIELDS]

rien (si rien tous les fields sont affiché par défaut)