mercredi 3 octobre 2018

Serveur de cartographie

Solutions existantes

- MapServer
- GeoServer
- OpenStreetMap (Gratuit)
- OpenMapTiles Map Server (Payant)
- OpenTileServer (Nécessite Ubuntu 14)
- QGIS (Nécessite python 3)
- Mapsurfer .NET (Créateur de carte)
- Google map (Payant)
- Stamen
- ….

Solution retenu : OpenStreetMap (Open source, libre, grande communauté, régulièrement mis à jour, disponible sous tout type d'OS). La majorité des autres solutions se base sur les données d'OSM.

L'installation a été réaliser sous Ubuntu Linux 18.04 LTS

Les dépendances de base à installer :

```
sudo apt install libboost-all-dev git-core tar unzip wget bzip2 build-essential autoconf libtool libxml2-dev libgeos-dev libgeos++-dev libpq-dev libbz2-dev libproj-dev munin-node munin libprotobuf-c0-dev protobuf-c-compiler libfreetype6-dev libtiff5-dev libicu-dev libgdal-dev libcairo-dev libcairomm-1.0-dev apache2 apache2-dev libagg-dev liblua5.2-dev ttf-unifont lua5.1 liblua5.1-dev libgeotiff-epsg curl
```

Dépendance à installer en global :

```
sudo npm install -g carto
```

Installation de PostgreSQL / Postgis

Sur Ubuntu, il existe des versions pré-packagées de Postgis et Postgresql :

```
sudo apt-get install postgresql postgresql-contrib postgis postgresql-10-postgis-2.4
```

Maintenant, il faut créer une base de données Postgis, avec son utilisateur. Le nom d'utilisateur devra être le même que celui qui fera le rendu de la carte avec Mapnik.

```
sudo -u postgres -i
createuser « utilisateur »
createdb -E UTF8 -O « utilisateur » « nom db »
```

Il faut maintenant aller sur psql et se connecter à la base de données créée avec les commandes :

```
psql
\c « nom db »
```

Créer les deux extensions :

```
CREATE EXTENSION postgis;
CREATE EXTENSION hstore;
```

Définir l'utilisateur créé comme propriétaire des tables

```
ALTER TABLE geometry_columns OWNER TO « utilisateur »;
ALTER TABLE spatial_ref_sys OWNER TO « utilisateur »;
```

Une fois fait on peut quitter psql et PostgreSQL.

```
\q
exit
```

Installation de osm2pgsql

osm2pgsql est un outils qui permet d'importer et de gérer les données d'OpenStreetMap dans une base de données. Nous allons le mettre dans un dossier à part.

```
mkdir ~/src && cd ~/src
git clone git://github.com/openstreetmap/osm2pgsql.git
cd osm2pgsql
```

Il va falloir installer d'autre dépendances:

```
sudo apt install make cmake g++ libboost-dev libboost-system-dev libboost-filesystem-dev libexpat1-dev zlib1g-dev libbz2-dev libpq-dev libgeos-dev libgeos++-dev libproj-dev lua5.2 liblua5.2-dev
```

On va créer un autre dossier « build », ou seront installer les composants d'osm2pgsql.

```
mkdir build && cd build
cmake ..
```

Une fois la commande fini :

```
make
```

Si la commande c'est bien exécuter elle se termine par : « [100%] Built target osm2pgsql »

Puis:

```
sudo make install
```

Installation de Mapnik

Mapnik est un mapping toolkit open source pour le rendu de carte utilisé par OSM.

Il suffit d'installer les dépendances :

```
sudo apt-get install autoconf apache2-dev libtool libxml2-dev libbz2-dev libgeos-dev libgeos++-dev libproj-dev gdal-bin libmapnik-dev mapnik-utils python-mapnik
```

Installation de mod_tile et rendred

Mod_tile à deux utilisation, la première, il s'agit d'un module personnalisé chargé de servir les tuiles et de demander le rendu des tuiles si elles ne sont pas déjà disponibles dans le cache ou si elles ont été modifiées depuis. La seconde, rendering back-end, qui gère les requêtes de mod_tile et qui est également responsable de la mise en file d'attente des demandes pour s'assurer que le serveur n'est pas surchargé si plusieurs demandes arrivent simultanément. Il existe actuellement deux implémentations d'un rendering back-end: renderd et tirex.

Il faut installer mod_tile dans le meme dossier que le module osm2pgsql:

```
cd ~/src
git clone -b switch2osm git://github.com/SomeoneElseOSM/mod_tile.git
```

Et nous allons générer le fichier:

```
cd mod_tile
./autogen.sh
```

Une fois fini la commande devrait afficher : « autoreconf: Leaving directory »

Ensuite il faut exécuter les commandes suivantes :

```
./configure

make

sudo make install

sudo make install-mod_tile

sudo ldconfig
```

Configuration du style de la map

Le style utiliser ici est celui utilisé par la carte «standard» sur le site openstreetmap.org. Il a été choisi parce qu'il est bien documenté et devrait fonctionner n'importe où.

On se rend dans la dossier ou sont installer osm2pgsql et mod_tile:

```
cd ~/src
git clone git://github.com/gravitystorm/openstreetmap-carto.git
cd openstreetmap-carto
```

Ensuite, on va installer le compilateur «carto».

```
sudo apt install npm nodejs
sudo npm install -g carto
```

Et enfin on va compiler le fichier project.mml en fichier xml utilisable par Mapnik:

```
carto project.mml > mapnik.xml
```

Télécharger les données / Cartes

Reste maintenant plus qu'a télécharger une carte et la rentrée dans la base de données, pour ca on peut aller sur :http://download.geofabrik.de/ par exemple. Attention la carte entière du monde peut prendre extrêmement longtemps à s'importer ! (48h ++)

Télécharger la carte de la Corse: http://download.geofabrik.de/europe/france/corse-latest.osm.pbf

On va créer un autre dossier pour stocker les données des cartes:

```
mkdir ~/data
cd ~/data
wget http://download.geofabrik.de/europe/france/corse-latest.osm.pbf
```

La commande suivante permet d'insérer les données d'OpenStreetMap dans Postgres. Il est possible de modifier leurs valeurs -C et -- number-processes en fonction de la RAM et du nombre de coeur disponibles:

```
osm2pgsql -d gis --create --slim  -G --hstore --tag-transform-script ~/src/openstreetmap-carto/openstreetmap-carto.lua -C 2500 --number-processes 1 -S ~/src/openstreetmap-carto/openstreetmap-carto.style ~/data/corse-latest.osm.pbf
```

Une fois fini, il va falloir dans le repertoire d'OpenStreetMap-carto afin de télécharger et indexer les bordures de la carte:

```
cd ~/openstreetmap-carto/scripts
./get-shapefiles.py
```

Cela peut prendre quelques temps en fonction de la carte téléchargé.

Attention certains noms de ville ou de rue ne sont peut être pas en latin, il faut donc télécharger des dépendances supplémentaires:

```
sudo apt-get install fonts-noto-cjk fonts-noto-hinted fonts-noto-unhinted ttf-unifont
```

Configuration de renderd

Il faut se rendre dans son fichier de conf:
`sudo nano /usr/local/etc/renderd.conf`

Certaines lignes peuvent être changer :

`num_threads=4 :`

A modifier on fonction du nombre de RAM disponible.

`XML=/home/« utilisateur »/src/openstreetmap-carto/mapnik.xml`

Vérifier que le chemin est correcte.

`URI=/hot/`

Il s'agit de l'url d'accès par default à la carte. Dans ma conf je l'ai modifié pour « osm_tiles ».

Configuration d'Apache

```
sudo mkdir /var/lib/mod_tile
sudo chown « utilisateur » /var/lib/mod_tile
```

```
sudo mkdir /var/run/renderd
sudo chown « utilisateur » /var/run/renderd
```

Il faut maintenant lié Apache et mod_tile, pour ca il va falloir créer le fichier suivant :
`sudo nano /etc/apache2/conf-available/mod_tile.conf`

Et y ajouter cette ligne :
`LoadModule tile_module /usr/lib/apache2/modules/mod_tile.so`

Il faut maintenant lié Apache et renderd, pour ça il faut modifier le fichier:
`sudo nano /etc/apache2/sites-available/000-default.conf`

Et y ajouter les lignes suivantes entre « ServerAdmin » et « DocumentRoot :

```
LoadTileConfigFile /usr/local/etc/renderd.conf
ModTileRenderdSocketName /var/run/renderd/renderd.sock
ModTileRequestTimeout 0
ModTileMissingRequestTimeout 30
```

Et redémarre le serveur apache2 (je le redémarre 2 fois parce qu'apparemment des fois il ne prend pas bien en compte les modifications):

```
sudo service apache2 reload
sudo service apache2 reload
```

Pour lancer le module « renderd »:

```
sudo -u 'username' renderd -f -c /usr/local/etc/renderd.conf
```

SI dossier manquant:

```
sudo mkdir /var/run/renderd
sudo chown 'username' /var/run/renderd
```

Rendu en arrière plan

Pour le faire tourner en arrière plan :

```
nano ~/src/mod_tile/debian/renderd.init
sudo cp ~/src/mod_tile/debian/renderd.init /etc/init.d/renderd
sudo chmod u+x /etc/init.d/renderd
sudo  cp  ~/src/mod_tile/debian/renderd.service  /lib/systemd/
system/
sudo /etc/init.d/renderd start
sudo systemctl enable renderd
```

Visualisation WEB

Pour visualisé la carte sur la navigateur il faut ensuite télécharger une librairie comme leafletJS :

```
cd /var/www/html
sudo wget http://cdn.leafletjs.com/leaflet/v1.2.0/leaflet.zip
sudo unzip leaflet.zip
```

Modifier le fichier index.html:

```
sudo nano /var/www/html/index.html
```

Pour qu'il ressemble à ca :

```
<html>
<head>
<title>My first osm</title>
<link rel="stylesheet" type="text/css" href="leaflet.css"/>
<script type="text/javascript" src="leaflet.js"></script>
<style>
   #map{width:100%;height:100%}
</style>
</head>

<body>
  <div id="map"></div>
  <script>
    var map = L.map('map').setView([53.555,9.899],5);
    L.tileLayer('http://your-ip/osm_tiles/{z}/{x}/{y}.png',{maxZoom:18}).addTo(map);
</script>
</body>
</html>
```

Enfin: « ip-du-serveur/index.html » pour visualiser la carte.