**A simple guide for MongoDB MapViewer**

**Introduction:**

MongoDB MapViewer is an application with friend GUI for importing, viewing and querying spatial data stored in MongoDB with the R-Tree module. The cross-platform design of MapViewer enables user to compile and run under in both Windows and Ubuntu.

**Compile from source:**

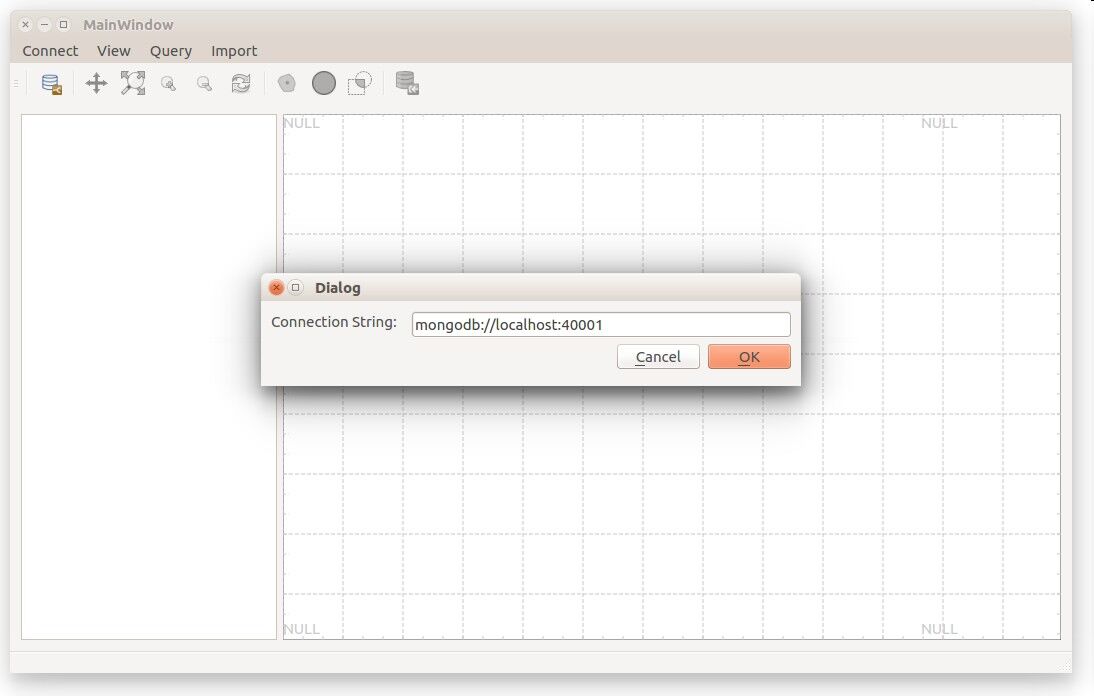
To compile MongoDB MapViewer, you might need the latest version of QT community IDE and additional libraries, which are:

1. MongoDB CXX Driver (legacy, master)
2. Boost Library 1.58 or higher version
3. GDAL library

**Usage:**

Before using MapViewer, please configure a MongoDB cluster: run shard servers (multiple) and config servers (one or three) with mongod, and run router server with mongos. And mongos should use the version compiled from the source code of MongoDB with R-tree support (<https://github.com/lmars-gis/mongo/tree/v3.2.0/rtree>). To help users to quickly start MapViewer, mongod, a windows binary files, i.e., mongod.exe, mongos.exe and mongo.exe, are place under the same path as MapViewer of windows version, i.e. “.\demo\compiled binaries\win 64”.

Then, you could execute MapViewer, and the first step is to connect the MongoDB cluster you just deployed, see the following picture. After that, the function icons will be enable, which means users could experience available functions, like view (including roam and zoom) and query (including range and near queries).



Linking to a MongoDB cluster

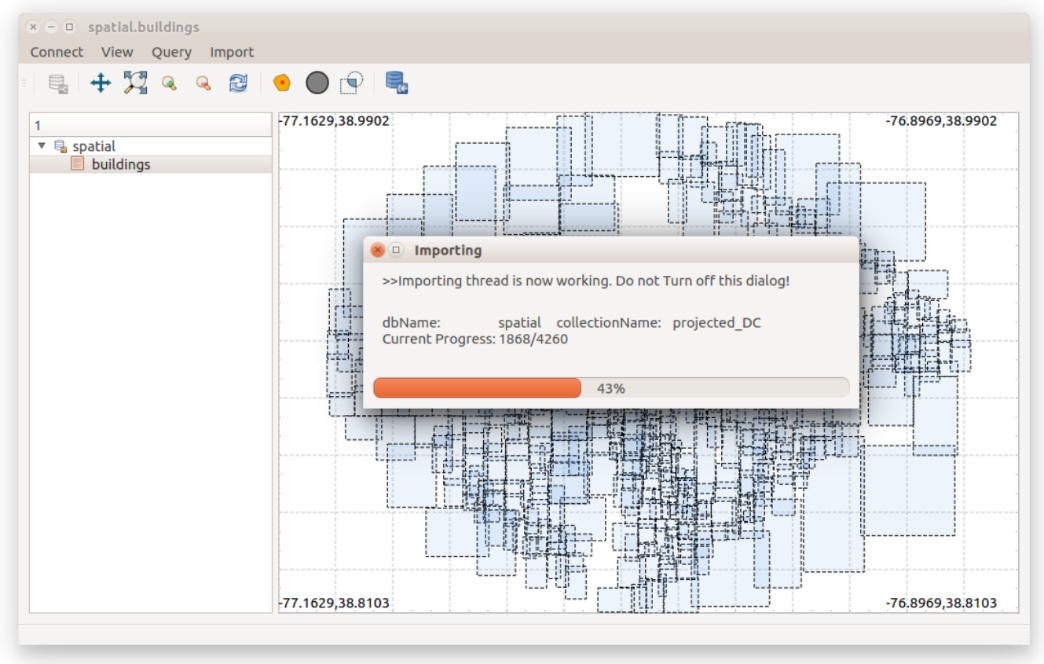
Next, users may import planar spatial data into the deployed MongoDB cluster. In the opened dialog, in addition to input database and collection to store a spatial dataset, the path to a shape file must be specified. Consequently, the process to load data and create R-tree will be started. When the task is finished, the imported spatial dataset will be displayed in the left window as a spatial layer. When clicking it, the hierarchical MBR information of the layer’s R-tree will be drawn in the right window.

We provided two sample spatial data of shape file format, which are located in “.\demo\compiled binaries\osm\_data”. The two datasets were downloaded from openstreetmap ([www.openstreetmap.org](http://www.openstreetmap.org)). Users could load them into the R-tree powered MongoDB, and then visualize and query them.

**Functions:**

1. **Import shape file directly into MongoDB**

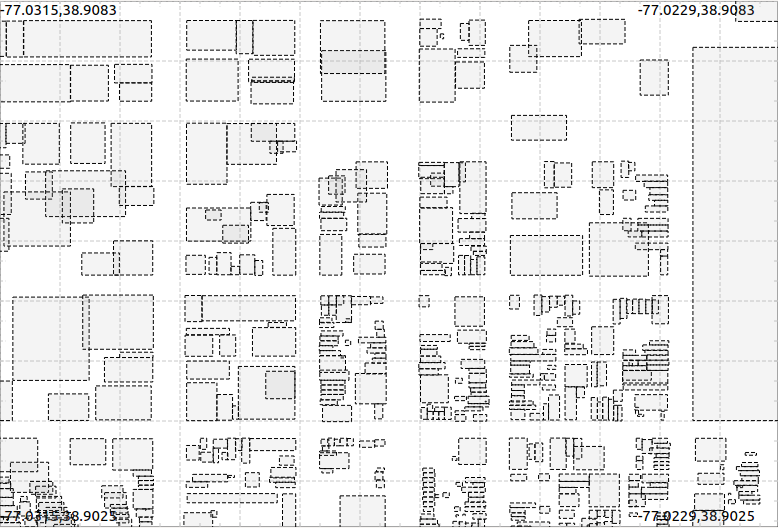
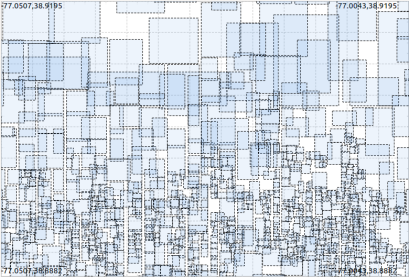
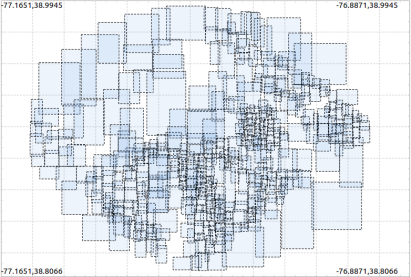
MongoDB MapViewer enables users to import ESRI shape file (\*.shp) directly into MongoDB database. If you import shape file data, MapViewer will firstly ask you to choose the geometry type so as to run registerGeometry command. Then metadata about geometries will registered, after which spatial data will be loaded and R-tree will be created.



Importing shape file within MapViewer

1. **Browse spatial data with LOD techs**

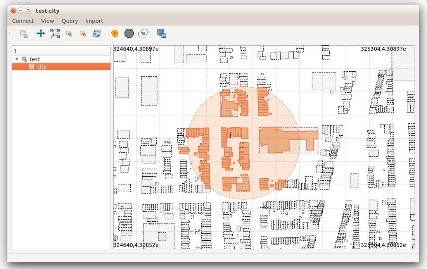
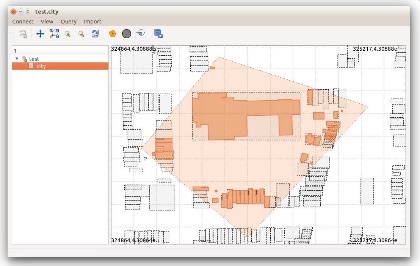
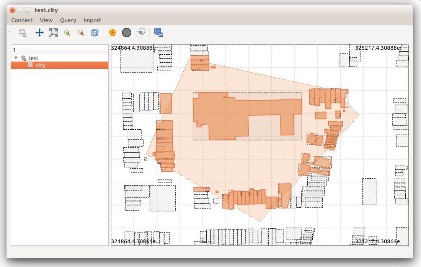
Facilitated by the multilevel structure of R-tree, we are able to draw MBRs of the spatial data using Level of Detail (LOD) techs. As we keep zooming in, more details of the spatial data coming up. If we reaches the data entry level, we will see MBRs in gray color. The LOD feature enables us to view and navigate large spatial dataset efficiently.



The LOD effect supported by MapViewer

1. **Query spatial data and visualize the results**

By implementing three spatial predicates ($geoWithin, $geoIntersects and $geoNear) based on R-tree index plugged into MongoDB, developers could query the spatial data through MongoDB driver (like in MapViewer). The following pictures show how MapViewer query and visualize spatial queries.



Range (left: using $geoIntersects, middle: using $geoWithin) and Near (right: using $geoNear) queries

**Note:** when performing queries, plase use mouse to draw a polygon (for $geoIntersects and $geoWithin) or circle (for $geoNear): click to fix points and double-click to finish geometry drawing.