MeraMap

Fast Track Deployment of customized OSM Tile Server

Parveen Arora
Dept. of Information Technology

July 30, 2011

Outline

1 Introduction
- Introduction To Openstreet Maps
- Why to Build Map for own?

2 Prerequisites
- Hardware requirements

3 Server Setup
- Setting up of own Server

4 Installation Terms
- PostGIS
- osm2pgsql
- Mapnik
- PostGIS with Mapnik
- Osmosis

5 MeraMap
- Objective

Outline

1. Introduction
   - Introduction To Openstreet Maps
   - Why to Build Map for own?
2. Prerequisites
   - Hardware requirements
3. Server Setup
   - Setting up of own Server
4. Installation Terms
   - PostGIS
   - osm2pgsql
   - Mapnik
   - PostGIS with Mapnik
   - Osmosis
5. MeraMap
   - Objective

Outline

1. Introduction
   - Introduction To Openstreet Maps
   - Why to Build Map for own?
2. Prerequisites
   - Hardware requirements
3. Server Setup
   - Setting up of own Server
4. Installation Terms
   - PostGIS
   - osm2pgsql
   - Mapnik
   - PostGIS with Mapnik
   - Osmosis
5. MeraMap
   - Objective

Outline

1. Introduction
   - Introduction To Openstreet Maps
   - Why to Build Map for own?
2. Prerequisites
   - Hardware requirements
3. Server Setup
   - Setting up of own Server
4. Installation Terms
   - PostGIS
   - osm2pgsql
   - Mapnik
   - PostGIS with Mapnik
   - Osmosis
5. MeraMap
   - Objective

Outline

1. Introduction
   - Introduction To Openstreet Maps
   - Why to Build Map for own?
2. Prerequisites
   - Hardware requirements
3. Server Setup
   - Setting up of own Server
4. Installation Terms
   - PostGIS
   - osm2pgsql
   - Mapnik
   - PostGIS with Mapnik
   - Osmosis
5. MeraMap
   - Objective

Introduction To Openstreet Maps

Openstreet Maps

OpenStreetMap is the editable World map of everything. It is the Wikipedia of maps. It is to other on-line maps as Wikipedia is to Britannica. And it is awesome in every possible way.

OpenStreetMap makes the data and the software available to you with Free Software and free data licenses so that you can use, learn from, teach with, improve upon and share with others what you gain from OpenStreetMap. And you can build your own local copy of OpenStreetMap for your business, school, community group or personal interests.

Introduction To Openstreet Maps

Openstreet Maps

OpenStreetMap is the editable World map of everything. It is the Wikipedia of maps. It is to other on-line maps as Wikipedia is to Britannica. And it is awesome in every possible way.

OpenStreetMap makes the data and the software available to you with Free Software and free data licenses so that you can use, learn from, teach with, improve upon and share with others what you gain from OpenStreetMap. And you can build your own local copy of OpenStreetMap for your business, school, community group or personal interests.

Why to Build Map for own?

Why indeed

OpenStreetMap.org is already freely available on the internet. Why not just use that? You can and you should. Eventually you may come up with an idea. You might want to make the map work a little differently for you. You might want a map for a special purpose.

Example

www.opencyclemap.org OpenCycleMap is a wonderful example of what you can do with the tools and data of OpenStreetMap catalyzed by an idea.OpenCycleMap uses OpenStreetMap data, then displays it in a way that is useful to cyclists with the emphasis placed on cycle trails, bike shops and bike parking

Why to Build Map for own?

Why indeed

OpenStreetMap.org is already freely available on the internet. Why not just use that? You can and you should. Eventually you may come up with an idea. You might want to make the map work a little differently for you. You might want a map for a special purpose.

Example

www.opencyclemap.org OpenCycleMap is a wonderful example of what you can do with the tools and data of OpenStreetMap catalyzed by an idea.OpenCycleMap uses OpenStreetMap data, then displays it in a way that is useful to cyclists with the emphasis placed on cycle trails, bike shops and bike parking

Hardware requirements

Bare minimum hardware requirements

- RAM: 1 GB
- Disk: 200 GB

Recommended minimum hardware requirements

- RAM: 4 GB
- Disk: 1 TB

Potential production hardware requirements

- RAM: 64 GB
- Disk: Many TB of 15,000 RPM RAID

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Hardware requirements

Bare minimum hardware requirements

- **RAM: 1 GB**
- **Disk: 200 GB**

Recommended minimum hardware requirements

- **RAM: 4 GB**
- **Disk: 1 TB**

Potential production hardware requirements

- **RAM: 64 GB**
- **Disk: Many TB of 15,000 RPM RAID**

Setting up of own Server

Creating tiles using Mapnik and generate_tiles.py

This is the method used by, for example, the OSM cycle map. Its main advantage is that nothing needs to run on the webserver - it just needs a directory of image files. So, for example, you can install all the software on your home PC, and transfer the tiles to your webhost when you're finished.

Setting up of own Server

Procedure

- Download the planet file from planet.openstreetmap.org
- Import into a PostGIS database using osm2pgsql
- Set up mapnik and test using osm.xml and the generate_image.py
- When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders
- Copy/move tiles into your webserver's document root.
- Change the OpenLayers instance to use your own tileserver instead of the main one

Setting up of own Server

Procedure

- **Download the planet file from planet.openstreetmap.org**
- Import into a PostGIS database using osm2pgsql
- Set up mapnik and test using osm.xml and the generate_image.py
- When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders
- Copy/move tiles into your webserver's document root.
- Change the OpenLayers instance to use your own tileserver instead of the main one

Setting up of own Server

Procedure

- Download the planet file from planet.openstreetmap.org
- Import into a PostGIS database using osm2pgsql
- Set up mapnik and test using osm.xml and the generate_image.py
- When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders
- Copy/move tiles into your webserver's document root.
- Change the OpenLayers instance to use your own tileserver instead of the main one

Setting up of own Server

Procedure

- **Download the planet file from planet.openstreetmap.org**
- **Import into a PostGIS database using osm2pgsql**
- **Set up mapnik and test using osm.xml and the generate_image.py**
- **When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders**
- **Copy/move tiles into your webserver's document root.**
- **Change the OpenLayers instance to use your own tileserver instead of the main one**

Setting up of own Server

Procedure

- **Download the planet file from planet.openstreetmap.org**
- **Import into a PostGIS database using osm2pgsql**
- **Set up mapnik and test using osm.xml and the generate_image.py**
- **When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders**
- **Copy/move tiles into your webserver's document root.**
- **Change the OpenLayers instance to use your own tileserver instead of the main one**

Setting up of own Server

Procedure

- Download the planet file from planet.openstreetmap.org
- Import into a PostGIS database using osm2pgsql
- Set up mapnik and test using osm.xml and the generate_image.py
- When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders
- Copy/move tiles into your webserver's document root.
- Change the OpenLayers instance to use your own tileserver instead of the main one

Setting up of own Server

Procedure

- Download the planet file from planet.openstreetmap.org
- Import into a PostGIS database using osm2pgsql
- Set up mapnik and test using osm.xml and the generate_image.py
- When everything works, use generatetiles.py to create 1000s of tiles in a special hierarchy of folders
- Copy/move tiles into your webserver's document root.
- Change the OpenLayers instance to use your own tileserver instead of the main one

PostGIS

Why to use PostGIS

PostGIS is an extension for the PostgreSQL database. PostgreSQL already has geometry types. PostGIS adds geospatial functions and two metadata tables which makes it pretty easy to handle spatial data in the database.

In the OSM project PostGIS is used for creating maps with Mapnik, many other people also use PostGIS for other tasks. The central OSM database was a MySQL database until API 0.5, since API 0.6 it is a PostgreSQL database, but currently not using the PostGIS extension.

PostGIS

Why to use PostGIS

PostGIS is an extension for the PostgreSQL database. PostgreSQL already has geometry types. PostGIS adds geospatial functions and two metadata tables which makes it pretty easy to handle spatial data in the database.

In the OSM project PostGIS is used for creating maps with Mapnik, many other people also use PostGIS for other tasks. The central OSM database was a MySQL database until API 0.5, since API 0.6 it is a PostgreSQL database, but currently not using the PostGIS extension.

osm2pgsql

osm2pgsql

osm2pgsql is a utility program that converts OpenStreetMap (.OSM) data into a format that can be loaded into PostgreSQL. It is often used to render OSM data visually using Mapnik, as Mapnik can query PostgreSQL for map data, but does not work directly with OSM files.

osm2pgsql is a lossy conversion utility. It only adds features that have certain tags, as defined in the config file, and it converts nodes and ways to linestrings and polygons. This means that you can't tell which linestring is connected to which, but for rendering a map that's not important

osm2pgsql

osm2pgsql

osm2pgsql is a utility program that converts OpenStreetMap (.OSM) data into a format that can be loaded into PostgreSQL. It is often used to render OSM data visually using Mapnik, as Mapnik can query PostgreSQL for map data, but does not work directly with OSM files.

osm2pgsql is a lossy conversion utility. It only adds features that have certain tags, as defined in the config file, and it converts nodes and ways to linestrings and polygons. This means that you can't tell which linestring is connected to which, but for rendering a map that's not important

Mapnik

What is Mapnik

Mapnik is the software we use to render the main Slippy Map layer for OSM, along with other layers such as the ”cycle map” layer and ”noname” layer. It is also the name given to the main layer, so things get a bit confusing sometimes.

Mapnik is a open source toolkit for rendering maps. It’s written in C++ and includes high-level Python bindings. It can read ESRI shapefiles, PostGIS, TIFF rasters, .osm files, any GDAL or OGR supported formats.

Mapnik

What is Mapnik

Mapnik is the software we use to render the main Slippy Map layer for OSM, along with other layers such as the "cycle map" layer and "noname" layer. It is also the name given to the main layer, so things get a bit confusing sometimes.

Mapnik is a open source toolkit for rendering maps. It's written in C++ and includes high-level Python bindings. It can read ESRI shapefiles, PostGIS, TIFF rasters, .osm files, any GDAL or OGR supported formats.

PostGIS with Mapnik

Use of PostGIS with Mapnik

The Mapnik renderer uses a PostGIS database as data source. OSM data is imported into PostGIS using the Osm2pgsql program. It contains the tables planet_osm_point, planet_osm_line, planet_osm_polygon for point, line and polygon features, respectively.In addition there is a table planet_osm_roads for some line features just shown on some zoom levels. Not all OSM data ends up in the database!

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Osmosis

Osmosis for OSM Data Handling

Osmosis is a command line Java application for processing OSM data. It has components for reading from database and from file, components for writing to database and to file, components for deriving and applying change sets to data sources, components for sorting data, etc

For Example:

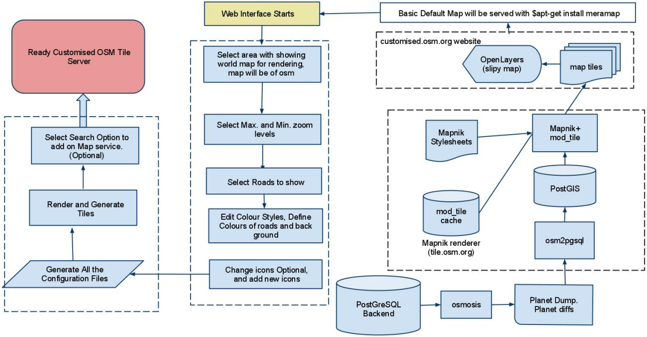
- **Generate planet dumps from a database.**
- **Load planet dumps into a database.**
- **Compare two planet files and produce a change set.**
- **Re-sort the data contained in planet files.**

Objective of MeraMap

Fast Track Deployment of OSM Tile Server

Fast Track Deployment of Customised OSM Tile Server" is a Google Summer of Code/2011 project by Parveen Arora also named as Mera Map at the early design stage. It will be one complete package that will automatically install all the components required to set up your own customised server which will update automatically with sync. of OpenStreetMap data for an area specified by implementor, It will also suggest him/her and will also give a option to choose default options i.e weekly bases , with easy user interface and browser based customisation of Map Style, adding of Icons, local language and additional search feature.

Working Model



Web Interface Starts

Basic Default Map will be served with $apt-get install meramap

customised.osm.org website

OpenLayers (slipy map)

map tiles

Select area with showing world map for rendering, map will be of osm

Select Max. and Min. zoom levels

Select Roads to show

Edit Colour Styles, Define Colours of roads and back ground

Change icons Optional, and add new icons

Ready Customised OSM Tile Server

Select Search Option to add on Map service. (Optional)

Render and Generate Tiles

Generate All the Configuration Files

Mapnik Stylesheets

Mapnik+ mod_tile

PostGIS

mod_tile cache

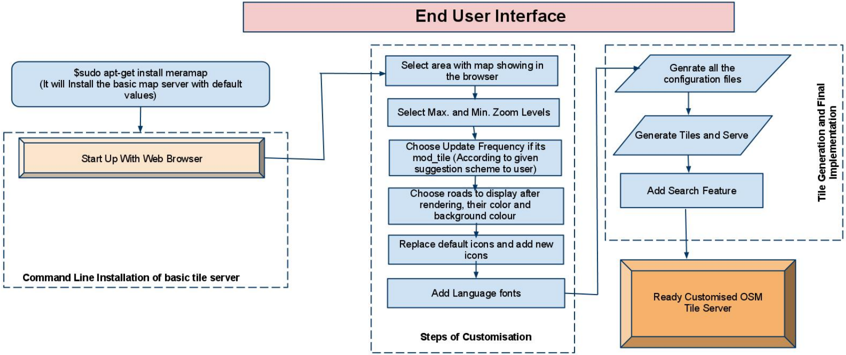
Mapnik renderer (tile.osm.org)

osm2pgsql

PostGreSQL Backend

osmosis

Planet Dump. Planet diffs



End User Interface

$sudo apt-get install meramap
(It will Install the basic map server with default values)

Start Up With Web Browser

Command Line Installation of basic tile server

Select area with map showing in the browser

Select Max. and Min. Zoom Levels

Choose Update Frequency if its mod_tile (According to given suggestion scheme to user)

Choose roads to display after rendering, their color and background colour

Replace default icons and add new icons

Add Language fonts

Steps of Customisation

Genrate all the configuration files

Generate Tiles and Serve

Add Search Feature

Tile Generation and Final Implementation

Ready Customised OSM Tile Server

Thank You

Openstreet Map Server
Parveen Arora
www.parveenarora.in

`Presentation made with` LaTeX