MapLocator

0.1

Installation and Configuration Guide

Prepared by

GS Lab

Great Software Laboratory Pvt. Ltd.
7th and 8th Floor, Amar Arma Genesis
Opp. Sapphire Chambers, Baner Road
Baner, Pune - 411 045
W: +91 20 4671 1000 / 4671 1100
Fax: +91 20 4671 1234
eFax: +1 425 963 4570
http://www.gslab.com

Table of Contents

1 System Requirements

This chapter covers the technical requirements necessary to install and configure the MapLocator.

Hardware Requirements

Minimal

1. Processor: 1GHz Standard Intel or equivalent
2. Memory: 256 MB
3. Disk: 10 GB
4. Network: 10/100 Mbps Ethernet Link

Recommended

1. Processor: Intel(R) Core(TM)2 Duo CPU E4600 @ 2.40GHz
2. Memory: 4GB DDR2
3. Disk: 80 GB Auto Incrementing
4. Network: 10/100 Mbps Ethernet Link

Software Requirements

Server Requirement

1. Operating System : Ubuntu Linux 8.10+
2. Web Server: Apache2
3. Scripting Languages: PHP5, Python
4. Database: PostgreSQL 8.3 along with Postgis plug-in
5. Mail Server: Postfix
6. Server Side Mapping Tools: Mapserver
7. Client Side Mapping Tools: OpenLayers, Mapfish
8. Other Software packages: GDAL, PHP5 GD module, PHP5 PostgreSQL module, PHP5 apache2 module, Subversion

Client Requirement

1. Web Browser: IE7+, Firefox3.0+
2. Browser Plug-in: Adobe Flash Player

2 Installing the Prerequisites

This chapter covers the instructions for preparing the machine with the software requirements..

Ubuntu Linux

Following are the instructions to install using the apt-get command on Ubuntu

1. Apache2: Open source web server used for serving MapLocator to client

- To install apache2 use the following command:

  ```
  $ sudo apt-get install apache2
  ```

- To start stop apache2 web server use the following commands:

  ```
  $ sudo /etc/init.d/apache2 start
  $ sudo /etc/init.d/apache2 stop
  ```

- To confirm if server is running open web browser on the server machine and go to http://localhost. An ordinary HTML page with "It Works!" displayed should be visible.
- To enable the rewrite module use the following command

  ```
  $ sudo a2enmod rewrite
  ```

2. PHP: Server-side HTML embedded scripting language

- To install PHP use the following command

  ```
  $ sudo apt-get install php5
  ```

- To install additional modules use the following command

  ```
  $ sudo apt-get install libapache2-mod-php5 php5-gd php5-pgsql
  ```

- To confirm if PHP is installed create test.php file in /var/www with the following PHP code

  ```
  <?php
  echo "PHP installed successfully";
  ?>
  ```

  Later open web browser and got to http://localhost/test.php. "*PHP installed successfully*" should be displayed

3. PostgreSQL: Sophisticated open-source Object-Relational DBMS

- To install PostgresSQL use the following command

  ```
  $ sudo apt-get install postgresql postgresql-client postgresql-contrib
  ```

- To install pgAdmin GUI application for working with the database

  ```
  $ sudo apt-get install pgadmin3
  ```

- To install PostGIS(spatial database extension for PostgreSQL)

  ```
  $ sudo apt-get install postgresql-8.3-postgis
  ```

- To change PostgresSQL admin user account password to 'postgres123' use the following commands in sequence

  -
    ```
    $ sudo su postgres -c psql template1
    ```

  -
    ```
    template1=# ALTER USER postgres WITH PASSWORD 'postgres123';
    ```

  -
    ```
    template1=# \q
    ```

  Do the same for the Unix user ‘postgres’

  -
    ```
    $ sudo passwd -d postgres
    ```

  -
    ```
    $ sudo su postgres -c passwd
    ```
    Enter ‘postgres123’ after the above command

- To start/stop/restart PostgresSQL server
  ```
  $ sudo /etc/init.d/postgresql-8.3 restart
  $ sudo /etc/init.d/postgresql-8.3 start
  $ sudo /etc/init.d/postgresql-8.3 stop
  ```

4. GDAL: Translator library for raster geospatial data formats

- To install gdal-bin use the following command
  ```
  $ sudo apt-get install gdal-bin
  ```

5. Postfix: Open source email server

- To install postfix use the following command
  ```
  $ sudo apt-get install postfix
  ```
  during installation select internet site as option of sending/receiving mails
- For changing any settings in postfix, edit /etc/postfix/main.cf

6. Subversion: An open-source revision control system

- To install subversion use the following command
  ```
  $ sudo apt-get install subversion
  ```

7. Map Server: Open Source platform for publishing spatial data and interactive mapping applications to the web

- To install map server and cgi script use the following command
  ```
  $ sudo apt-get install cgi-mapserver mapserver-bin mapserver-doc
  ```
- To install additional supporting packages for PHP and Python
  ```
  $ sudo apt-get install php5-mapscript python-mapscript
  ```

8. Python: Interpreted, interactive, object-oriented, extensible programming language

- To install python use the following command
  ```
  $ sudo apt-get install python
  ```

9. OpenLayers: Makes it easy to put a dynamic map in any web page

- Download latest version of OpenLayers from http://openlayers.org and extract it to 'openlayers' directory
- To allow http access to OpenLayers edit the /etc/apache2/apache2.conf and append the following lines

  ```
  Alias /openlayers/ "<path to OpenLayers directory>"
  <Directory "<path to OpenLayers directory>">
   AllowOverride None
   Options Indexes FollowSymLinks Multiviews
   Order allow,deny
   Allow from all
  </Directory>
  ```
- To confirm OpenLayers is installed open browser on server and go to http://localhost/openlayers/. The contents of the OpenLayers directory should be displayed

10. MapFish: Flexible and complete framework for building rich web-mapping applications

- Download latest version of MapFish from http://mapfish.org/ and extract it to 'MapFish' directory
- To allow http access to MapFish edit the /etc/apache2/apache2.conf and append the following lines

  ```
  Alias /MapFish/ "<path to MapFish directory>"
  <Directory "<path to MapFish directory>">
   AllowOverride None
   Options Indexes FollowSymLinks Multiviews
   Order allow,deny
   Allow from all
  </Directory>
  ```
- To confirm MapFish is installed open browser on server and go to http://localhost/MapFish/. The contents of the MapFish directory should be displayed

3 Deploying the system

This chapter covers the instructions for deploying MapLocator.

Ubuntu Linux

Following are the instructions to deploy the system on Ubuntu Linux

Using the deployment script

1. Create the following directories tree structure under “~/”

```
MapLocator_deployment_files
        ├───source # to store the source code
        ├───backup # to store the backup
        ├───deploy # to deploy the system
        │   ├───MapFish # MapFish deployment
        │   ├───openlayers # OpenLayers deployment
        │   └───maplocator # MapLocator deployment
        ├───deployment_scripts # to store the deployment scripts
        └───temp #used as temp storage by deployment script
```

2. Download and extract the source code to source folder from: http://github.com/maplocator/maplocator/archives/master
3. Copy the deployment scripts to the deployment_scripts folder. The deployments scripts will be located at “trunk\scripts\deployment” in the source directory
4. Edit the deploy_local.php file located in deployment_scripts folder and make the following changes
    a. Set “source_path” to path of the source directory
    b. Set “deploy_path” to path of maplocator directory under the deploy folder as mentioned in the above tree structure
    c. Set “backup_path” to path of the backup directory
5. The final directory tree structure should look as follows:

```
MapLocator_deployment_files
        +---backup
        +---deploy
        |  +---MapFish
        |  +---OpenLayers
        |  \---maplocator
        +---deployment_scripts
        |  |  deploy_maplocator.py # python deployment script
        |  |  maplocator.yaml # configuration file
        |  |  README.txt # simple readme on how to use
        |  \---yaml
        +---source
        |  \---trunk        #
        |     +---config    #
        |     +---docs      #          MapLocator Source Code
        |     +---lib         #
        |     +---scripts     #
        |     \---source      #
        \---temp
```

6. Change path to deployment scripts directory and execute the following command:

   $ php deploy_local.php

   - If there are no errors in the execution the system should be deployed under “maplocator” directory. To confirm check the “maplocator” directory under deploy folder.

Creating and Setting up the database

1. **Creating a postgis template** (required for first time installation)

   The following shows how to create a new template. This is required so that every time a new database is created with this template, the ‘spatial_ref_sys’ and ‘geometry_columns’ tables are created by default. ‘spatial_ref_sys’ table contains all the supported projection systems. ‘geometry_columns’ table contains the list of layer tables, their projection (as srid), geometry column name, geometry type (point, polygon, line etc.).

   - Run the following commands in order to create the template:

     $ psql template1

     # \c template1

     # CREATE DATABASE template_postgis WITH template = template1;

   - Set the ‘datistemplate’ record in the ‘pg_database’ table for ‘template_postgis’ to TRUE indicating it’s a template

     # UPDATE pg_database SET datistemplate = TRUE WHERE datname = ‘template_postgis’;

     # \c template_postgis

     # CREATE LANGUAGE plpgsql ;

     # \i /usr/share/postgresql-8.3-postgis/lwpostgis.sql;

     # \i /usr/share/postgresql-8.3-postgis/spatial_ref_sys.sql;

   - In a production environment it could be a necessity to give role based permissions, but granting all for now.

     # GRANT ALL ON geometry_columns TO PUBLIC;

     # GRANT ALL ON spatial_ref_sys TO PUBLIC;

   - Vacuum freeze: it will guarantee that all rows in the database are "frozen" and will not be subject to transaction ID wraparound problems.

     # VACUUM FREEZE;

2. **Creating and Patching MapLocator database**

   - Create database (say for ex. maplocator) using the following commands

     $ psql -U postgres

     # CREATE DATABASE MapLocator WITH template= template_postgis ENCODING = ‘UTF8’; (Database MapLocator is created with two tables geometry_columns and spatial_ref_sys)

```
# \q
```

- Run the "maplocator.sql" script located in the maplocator source under "source\trunk\scripts\database" for the maplocator database using the following command

```
$ psql -U postgres maplocator < maplocator.sql
```

Post deployment configuration

1. Editing apache configuration file

Ideally the deployment directory for MapLocator can reside anywhere on the hard disk in which case, an "Alias" has to be created in Apache which would point to this location.

- To allow http access to MapLocator edit the /etc/apache2/apache2.conf and append the following lines to the end

```
Alias /maplocator/ "<path to MapLocator deployment directory>"
<Directory "<path to MapLocator deployment directory>">
        AllowOverride None
        Options Indexes FollowSymLinks Multiviews
        Order allow,deny
        Allow from all
</Directory>
```

- Restart Apache2 webserver using the following command:

```
$ sudo /etc/init.d/apache2 restart
```

2. Editing settings.php of the deployment

This file will be located in the deployment directory under 'sites/default' directory.

- Create copy of 'settings.php.default' as 'settings.php'

```
$ cp default.settings.php settings.php
```

- Uncomment and change the following line in 'settings.php' file

```
$db_url = 'pgsql://username:password@localhost/databasename';
```

  Set proper username, password and databasename as mentioned while creating the 'maplocator' database

3. Creating directories and setting permissions

- Create the following directories under the deployment root
  - shapefiles
  - upload
  - sites/default/files
- Set apache user (www-data) owner for the directories: upload, sites/default/files under the deployment directory using the following command:

```
$ sudo chown -R www-data:www-data shapefiles sites/default/files upload
```

4. Edit the .htaccess file (Should be done for every deployment)

- Edit the .htaccess, uncomment and change the "RewriteBase /" to "RewriteBase /maplocator"

5. Sign Up for the Google Maps API and add it to includemap.js

- Generate google maps api key for your domain/ipaddress using the following link ‘http://code.google.com/apis/maps/signup.html‘
- Copy the api key make the following changes to the file “sites/all/modules/map/includemap.js” under the deployment root.

  Under the switch statement *"switch (BASE_MAP_SOURCE_sp[i]) case ‘GOOGLE’:"* add the following code snippet after adding the Google Maps API Key:

  ```
  else if((document.location.href).indexOf("http://<IP or URL of website >") > -1) {
      document.write("<script src=‘http://maps.google.com/maps?file=api&amp;v=2&amp;key=<Google Maps API Key >‘ type=‘text/javascript’></script>");
    }
  ```

  Within the if statement ‘if(GOOGLE_EARTH_ENABLED)’ add the following code snippet:

  ```
  else if((document.location.href).indexOf("http://< IP or URL of the  Website >") > -1) {
      document.write("<script src=‘http://maps.google.com/jsapi?key=<Google Maps API Key >‘ type=‘text/javascript’></script>");
    }
  ```

6. Generate Map Files and copying the dependencies to cgi-bin

- The scripts to generate map files are located at “trunk\scripts\generateMapfiles” under the source root. To generate the .map files for line and polygon layers use the following command:

  ```
  $ php generateMapfiles.php -u DBUser -d DBName -p DBpassword
  ```

- Once you execute the above command files with .map extension will be generated under the generateMapfiles directory. Copy this file to cgi-bin directory using the following command

  ```
  $ sudo cp *.map /usr/lib/cgi-bin/
  ```

- Copy the examples.sym file to cgi-bin

  ```
  $ sudo cp examples.sym /usr/lib/cgi-bin/
  ```

  Once you perform the above steps all the line and polygon layers will be visible on the map.

4 Accessing and managing the system

This chapter covers various options for managing the system and information about accessing the system

1. To access the system open web browser from any client machine connect to the deployed system using a standard network. Once the browser is opened you can access MapLocator using the following url:

   http://<ip_or_hostname>/maplocator
2. Following are the credentials or admin access:
   - Username: admin
   - Password: m@pl0c@t0r
3. The administration dashboard is available at the following url:

   http:// <ip_or_hostname>/maplocator/admin
4. Drupal modules can be manages at the following url:

   http:// <ip_or_hostname>/maplocator/admin/build/modules
5. There are couples of custom drupal modules created for MapLocator. Following is the list:
   1. map
   2. node_mlocate_feature
   3. node_mlocate_layerinfo
   4. node_mlocate_themeinfo
6. Drupal theme can be manages at the following url:

   http:// <ip_or_hostname>/maplocator/admin/build/themes

Further information regarding the usage and design of the system is available at the following path under source: trunk/docs