## OSGIS - INTRODUCTION 2 (13.10.18)

<https://etherpad.net/p/HsKA-OSGIS>

**Bug trackers**

**Qgis**

2009 Hanover v1.0- more analytic functions

Almost all of the plugins written in python

Subversioning: GIT

**GRASS**

Old version of subversion (SVN)

**GIT (new version/style of subversion)**

Next wednesday introduction

**OSGeo Live**

[OSGeo Live](http://live.osgeo.org/) is a self-contained bootable DVD, USB thumb drive or Virtual Machine

**Data Storage Strategy**

**Files**

Shape

**DB**

PostGIS - addon to postgresql

MySQL - not many people store geo data

Spatialite - DB in one file, easy to share.

**Services (no storage)**

OGC (WMS, WFS, WCS, TMS, CSW, WPS, ...)

WMS cannot calculate buffer, unlikely in WFS (because it’s an image ;) ). (not geo Image)

WFS Format: geoJSON or gml - depending on your request

WMS Format: image formats - JPG (no transparency, not appropriate for point layer), PNG, TIFF

WCS - Raster Data, RAW Data. Calculations can be done on.

CSW (catalog service for the web) for what? (The catalogue is made up of records that describe geospatial data (e.g. [KML](https://en.wikipedia.org/wiki/Keyhole_Markup_Language)), geospatial services (e.g. [WMS](https://en.wikipedia.org/wiki/Web_Map_Service)), and related resources.)

WPS - web processes

**Data acquisitions and help projects/libraries (all written in C++)**

* GDAL(raster) - read from WMS and save it as geo JPG/ OGR(vector) - read WFS (geojson, gml) and save it to shape, in general conversion library. Geo\*\*\*\* () does not need to worry about formats
* PROJ4 - functionalities to coordinate transformations, based on EPSG database, (ex:Qgis doesn't do coordinate transformation but uses PROJ4)
* GEOS (Geometry Engine - Open Source) - provides geometrical operations (showing, labeling, implementing simple feature formats, all overlay operations)

JAVA Library

* GeoTools - For JAVA , encompasses most of the above libraries

**Service oriented architectures**

* GeoServer(Tomcat)/MapServer - all in all rendering engines

OGC WS are working principles. Not only open source SW.

Necessary functionalities:

* Read geo data (with gdal/ogr)
* Reproject if needed (proj4)
* Style (i.e. how should be points presented, if we request a WMS: circles, stars etc.),GeoServer uses **SLD**, in MapServer you have to write it on your own
* read/return request (http (Web Server))
* Understand cgi ?

Client (JS libraries & runs in internet browser, therefore not spatially enabled)

* OpenLayer
* Leaflet

Images reprojected/projected in server side. Only on HTML5 there are possibilities that might allow to do projection on client side, still not on each browser.

**Practical Part (Virtual Machine)**

Mapnik - creating tiles from OSM

**MapProxy** Proxy servers can be used to reproject respond of WMS#

Linux Small commands (all file reading begins with slash):

**pwd** - print working directory

**dir -** directories

**cd** -change directories

**cd** .. /.. → goes up, dot stands for current directory, .. for a folder up

**tab -** browses through the directories in the root folder

Programm name **-h** → provides help on the program (i.e . **shp2pgsql (-h)**)

**passwd -** changes pass

**history -**  to see command history

**clear -** doesn’t clear history, just the terminal

**sudo -** super user rights (sudo apt update)

Ls -

**chmod -** changes mode on accessing the file (i.e. chmod a+x proxyon.\*) (change mod)

**chown** -R tomcat8: geoserver - change owner

-R → recursive

**systemctl start tomcat8**

ll \*.shp - lists all shapefiles in the directory (long listing)

ll \*.shp | wc -l → number of lines

createdb -E UTF-8 -O user <mydb\_name>

Psql -l → list databases

Psql -d → write sql command. Ends with semicolon

Echo “CREATE EXTENSION postgis;” | psql - d <mydb\_name>

For i in \*.shp; do echo $1; done

mkdir <dirname> - creates new directory

**Import shape to pg**

shp2pgsql - s <sird> -g <geocol> -I <myshp> <mytable>

It is small tool reading the shape file reading from the attribute table (name/geom/)

**shp2pgsql** **-s** *4326* **-g** *geom* **-I** *gis\_osm\_natural\_free\_1.shp trees* → read shape file

**for** i **in** \*.shp **; do** **shp2pgsql -I -s** 4326 $i `**basename** $i .shp` **| psql** **-d** <mydb\_name> ; done

**PostgreSQL/PostGIS**

* Has its own user management different from OS, obvious :/
* Many database, each with many users, each user with specific permissions
* In command line psql
* pqRoute is extension on postGIS

**References**

<https://download.geofabrik.de/europe/germany/baden-wuerttemberg.html>

geoTIFF,shapefile, geoJSON

## Lesson 3, 17/10/2018

Review

* Pgsql → postgis
* Adding shapefiles to the DB (from Geofabrik)
* Run GeoServer
* Connect geoServer with DB

- read Geodata source (styling)

- didn’t create own style (today)

(OGC standards very important to Styled Layer Descriptor (SLD) ) **(important in project work and Exam)**

To see methods-> inspect element: Get, Post..

GEOPORTAL - example 1/ <https://www.imis.bfs.de/geoportal/>

* Mouse hover → GetFeatureInfo() (OGC)
* Graph was created D3.js
* How to know requests as an User:
  + right click - inspect the element
  + Developer console (F12) - Network Analysis

-CSW catalog service Catalogue services support the ability to publish and search collections of descriptive information (metadata) for data, services, and related information objects.

:

Example 2/ OSRM Open Source Routing Machine [project-osrm.org/](http://project-osrm.org/)

Clone code from github

* Start server, (What is 0.0.0.0 (means everywhere any computer 127.0.0.1 or 'localhost', whatever that means) port 8000)
* Minify JavaScript or CSS code Make your website smaller and faster to load
* Some company uses Graph Hopper (<https://www.graphhopper.com/>) or LBS, the answer is json

Some more examples:

* <https://openrouteservice.org/>
* <http://geojsonlint.com/> (to check if geoJSON is valid) (lint for everything could be find, i.e. jslint….)
* Some standards are not in OGS, ( <https://github.com/OSGeo/ows.js> )
* As far as he knows leaflet doesn’t have WPS support. Again whatever that means

Useful codes

\*python -m SimpleHTTPServer (in the cloned/copied folder) and open http://localhost:\*/public/

\* turf.js useful library Advanced geospatial analysis for browsers and Node.js.

\* EXt.js <https://www.sencha.com> (to make windows on the page fancy)

\* <https://gruntjs.com/>

\* <https://bower.io/>

Format to describe module ?

Example3: <http://brouter.de/brouter-web>

Example 4: Geo Vogatic

Objectives:

* To see requests and responses and analyse responses

**Part 2**

**\* sudo adduser user users (sudo adduser <USER> <GROUP>)**

**\* sudo systemctl start tomcat8**

**\* what kinds of data we can use in geoportal ?**

**\* Shape - image - tif -**

**\* SLD is XML language**

**\* How color format (#AAAAA) works? Hex color**

Customize Styles

* Style content (can copy existing style):
  + Color manipulation
  + Border style
* SLD cookbook (https://docs.geoserver.org/stable/en/user/styling/sld/cookbook/)

**Part 3**

**OpenLayers**

## OSGIS (07.11.2018)

OSTaps - Web client for choropleth maps

Creating choropleth map from geoJSON and Shape files

>DB - django - request with

What about the projection? (equal area?)

EPSG:3857 → WGS 84 / Pseudo-Mercator -- Spherical Mercator, Google Maps, OpenStreetMap, Bing, ArcGIS, ESRI

(S)IGNIS

DOOM - trash repository

Github

* git init
* git add -A
* git add <filename>
* git commit -m ‘comment’
* git branch my\_branch
* git checkout my\_branch
* git merge my\_branch
* git status
* git log
* git remote add origin <https://github.com/Agageldi/test.git>
* git push -u origin master