L04

Introduction to Geoserver: Publishing, Managing and Accessing geospatial data Services

Rahul Kumar
Research Scientist

NERDRR,NESAC

What is Geoserver?

Geoserver Overview

- An **open-source server** designed for **sharing, processing, and visualizing geospatial data**.
- Built using the **Java programming** language.
- First released in 2001 and actively maintained by a global developer community.

Key Features

- Supports **publishing geospatial data** from a variety of data sources (e.g., PostGIS, Shapefiles, GeoTIFF, etc.).
- Enables **web-based visualization** of spatial data through interactive maps.
- Allows **interoperability** with other GIS tools via **open standards**.

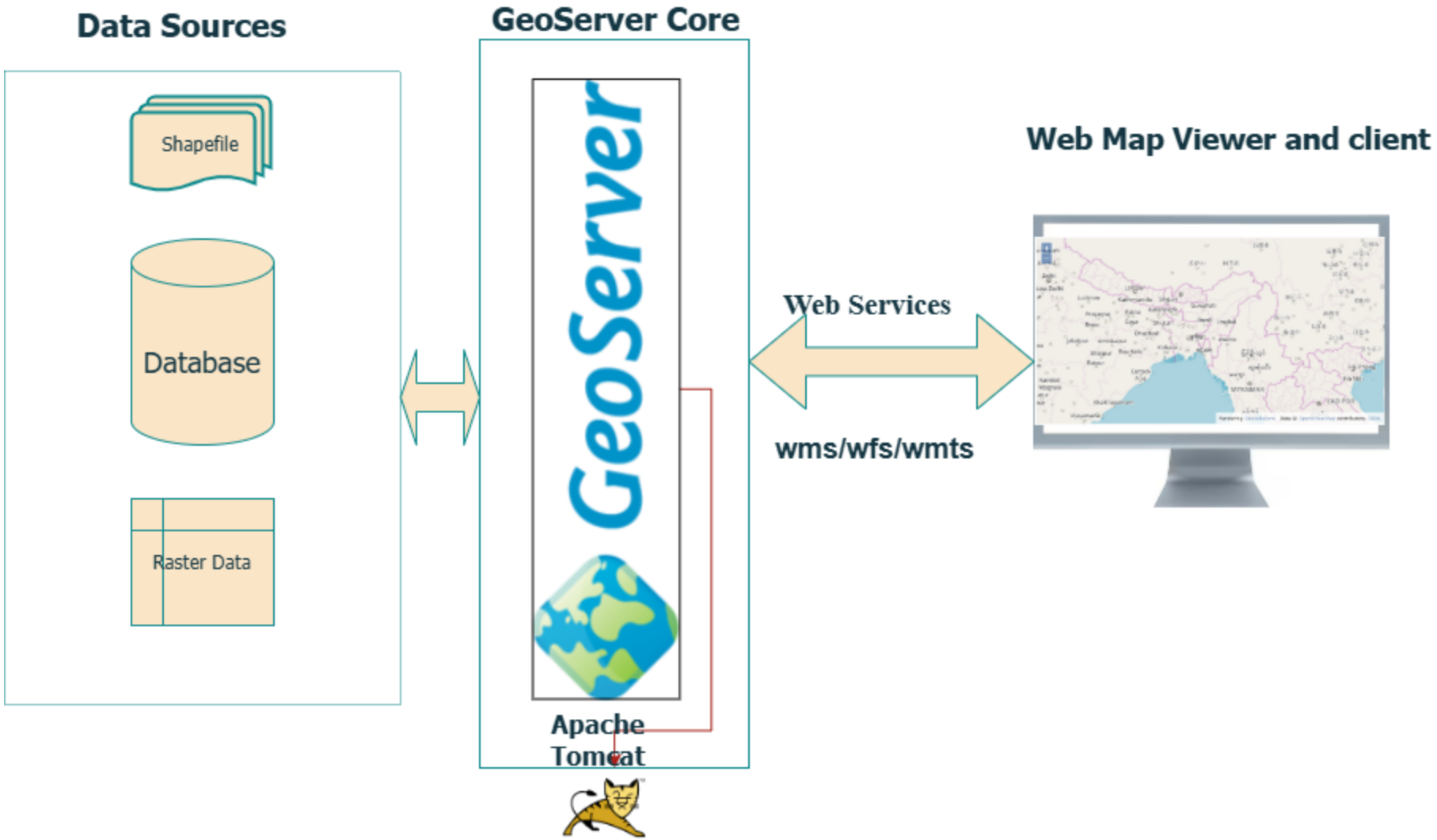
Open Standards Supported (OGC Standards)

- **WMS (Web Map Service):** Serves maps as images (PNG, JPEG, etc.) or vector tiles.
- **WFS (Web Feature Service):** Provides access to raw vector data for analysis and querying.
- **WCS (Web Coverage Service):** Shares raster data (e.g., satellite imagery, elevation models).
- **WMTS (Web Map Tile Service):** Serves tiled map data for faster rendering.

Why Geoserver?

- **Free and open-source**, reducing cost barriers.
- Seamlessly integrates with **web-based mapping clients** like OpenLayers, Leaflet, and QGIS.
- Handles **styling of maps** using Styled Layer Descriptor (SLD) and CSS-like formats.
- Provides robust tools for **filtering and querying data**, enabling dynamic map interaction.
- Easy to deploy on a wide range of platforms (Windows, Linux, macOS).
- Backed by a strong **community and ecosystem** for support and extensions.

GeoServer Architecture



GeoServer Architecture

1. Data Sources

- GeoServer supports a variety of data sources, including:
  - Spatial files (e.g., Shapefiles, GeoTIFF).
  - Databases (e.g., PostGIS, Oracle Spatial).
  - Web Feature Services (WFS) and Web Coverage Services (WCS).
  - Other formats like GeoPackage, KML, and JSON.

2. GeoServer Core

- Acts as the central processing unit of GeoServer.
- Responsible for:
  - Reading and interpreting spatial data.
  - Managing data stores.
  - Configuring services like layers, styles, and projections.
- Integrates with extensions for advanced functionality, such as raster processing or vector transformations.

3. Web Services

- Provides services for accessing and interacting with spatial data:
  - Web Map Service (WMS): Renders spatial data as map images (PNG, JPEG).
  - Web Feature Service (WFS): Serves raw vector data (e.g., GeoJSON, GML).
  - Web Coverage Service (WCS): Offers access to raster data (e.g., elevation, imagery).
  - Other APIs like WPS (Web Processing Service) for geospatial analysis.

4. Web Map Viewer and client

- A browser-based interface (e.g., GeoServer's preview feature) for viewing and testing published layers.
- Allows users to interact with and query the data visually.
- Applications that consume GeoServer outputs, such as:
  - GIS software (e.g., QGIS, ArcGIS).
  - Web mapping libraries (e.g., OpenLayers, Leaflet).
  - Custom-built web or mobile applications.

GeoServer Installation

1. Prerequisites:

- **Java Development Kit (JDK):** Ensure you have a compatible JDK installed and set up your environment variables correctly.
- **Apache Tomcat:** Download and install the appropriate version of Tomcat for your operating system.
- **GeoServer WAR file:** Download the latest stable release of the GeoServer WAR file from the official GeoServer website: https://geoserver.org/download/

2. Deploy the GeoServer WAR file:

- **Stop Tomcat:** If Tomcat is running, stop the service.
- **Locate the `webapps` directory:** Find the `webapps` directory within your Tomcat installation.
- **Copy the WAR file:** Copy the downloaded `geoserver.war` file to the `webapps` directory.
- **Start Tomcat:** Start the Tomcat service.

3. Verify GeoServer Deployment:

- **Access GeoServer:** Open a web browser and navigate to: http://localhost:8080/geoserver (or the appropriate hostname and port if you're using a different configuration).

https://nerdrr.gov.in/geoserver/web/wicket/bookmarkable/org.geoserver.web.demo.MapPreviewPage;jsessionid=F4F28C6DEA3DFD332F313F4BD09C92E7?0&filter=false

GeoServer

username password Remember me Login en

About & Status

About GeoServer

Data

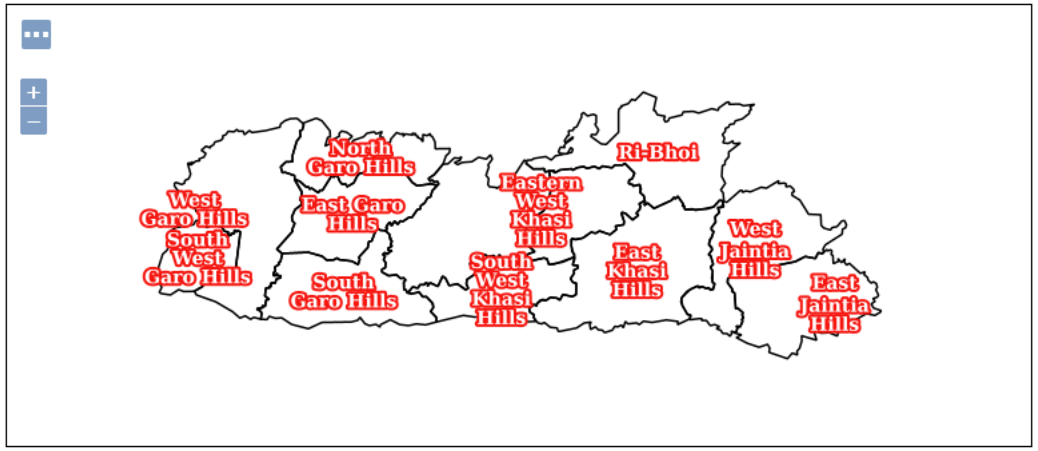
Layer Preview

List of all layers configured in GeoServer and provides previews in various formats for each.

<< < 4 5 6 7 8 > >> Results 126 to 150 (out of 581 items)

Search

https://nerdrr.gov.in/geoserver/forestfiremegh/wms?service=WMS&version=1.1.0&request=GetMap&layers=forestfiremegh%3Amg_district&bbox=89.8212356567383%2C25.0230560302734%2C92.80481719

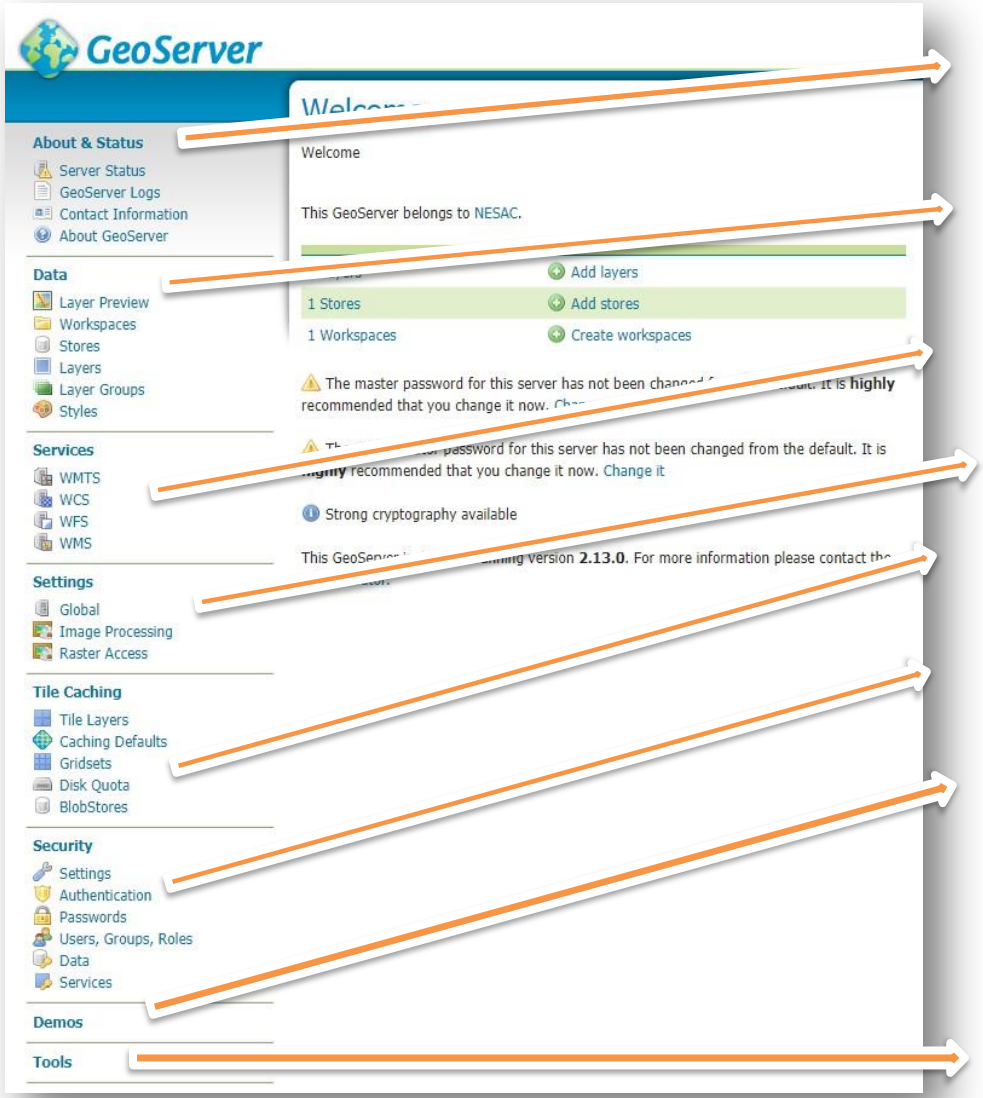


Scale = 1 : 2M 92.32361, 25.70251

Click on the map to get feature info

lightning_data_lines	nerdrrlightningWS:lightning_data_lines	OpenLayers GML KML	Select one
BT	cloudWS:BT	OpenLayers KML	Select one
NER_STATES	workgroup:NER_STATES	OpenLayers GML KML	Select one
NER_WATERBODY_10K_298	workgroup:NER_WATERBODY_10K_298	OpenLayers GML KML	Select one
gpm_ner_grid	workgroup:gpm_ner_grid	OpenLayers GML KML	Select one

Understanding GeoServer : Web administration interface



The **About & Status** section provides access to GeoServer **diagnostic and configuration tools**, and can be particularly useful for debugging.

The **Data** management section contains configuration options for all the **different data-related settings**.

The **Services** section is for **configuring the services** published by GeoServer.

The **Settings** section contains **configuration settings** that apply to **the entire server**.

The **Tile Caching** section configures the embedded **GeoWebCache**.

The **Security** section configures the **built-in security** subsystem.

The **Demos** section contains links to example WMS, WCS, and WFS requests for GeoServer as well as a listing all SRS info known to GeoServer. In addition, there is a reprojection console for converting coordinates between spatial reference systems, and a request builder for WCS requests

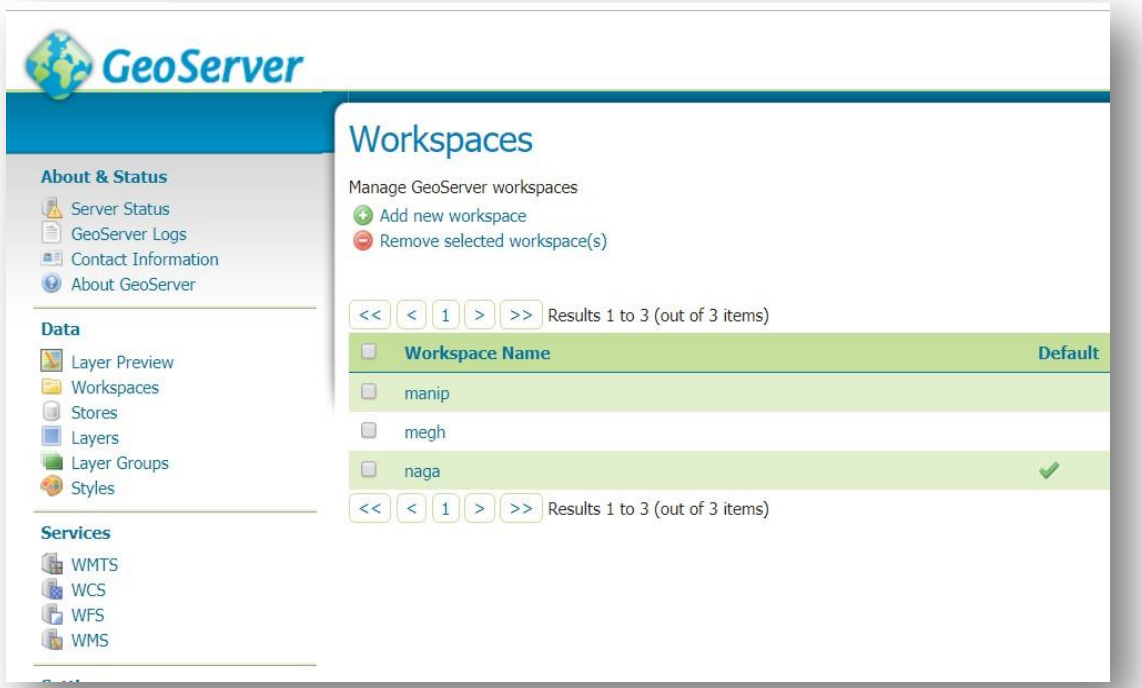
The **Tools** section **contains administrative tools**. By default, the only tool is the Catalog Bulk Load Tool, which can bulk copy test data into the catalog.

Publishing Data in GeoServer

Eg. Publishing a Shapefile

1. Create a new workspace
    1. In a web browser, navigate to http://localhost:8080/geoserver
    2. Login and Navigate to **Data ▸ Workspaces**.

A workspace is often used to group similar layers together. Layers may be referred to by their workspace name, colon, layer name (for example **naga**:circle_boundary).

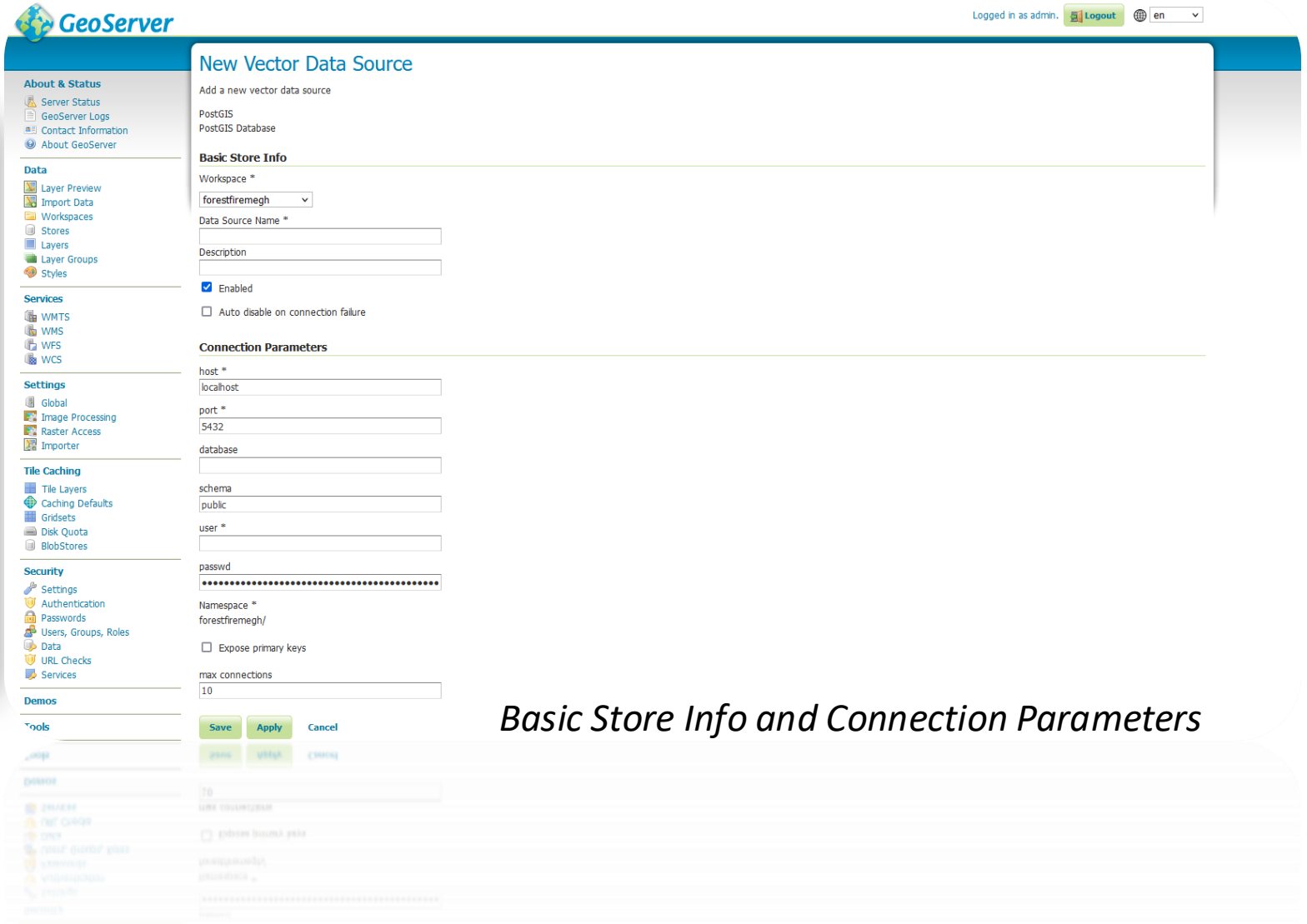


Publishing Data in GeoServer

2. Create a Store under the workspace. The store tells GeoServer **how to connect to the shapefile.**
    1. Navigate to **Data▸Stores**.
    2. You should see a list of stores, including the type of store and the workspace that the store belongs to.
    3. In order to add the shapefile, you need to create a new store. Click the **Add new Store** button. You will be redirected to a list of the data sources supported by GeoServer.
    4. Click **Postgis**. The **New Vector Data Source** page will display.
    5. Begin by configuring the **Basic Store Info**.
        - Select the workspace **naga** from the drop down menu.
        - Enter the **Data Source Name** as Nag Boundary
        - Enter a brief **Description** (such as “Nagaland Circle Boundary”).
    6. Under **Connection Parameters**, Provide the credential of database to fetch data,
    7. Click **Save**. You will be redirected to the **New Layer** page in order to configure thelayer

Publishing Data in GeoServer

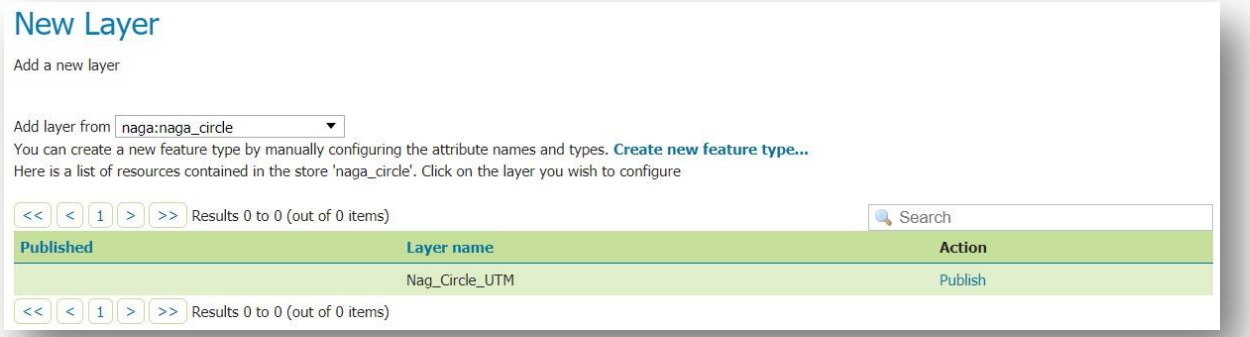
Data Stores



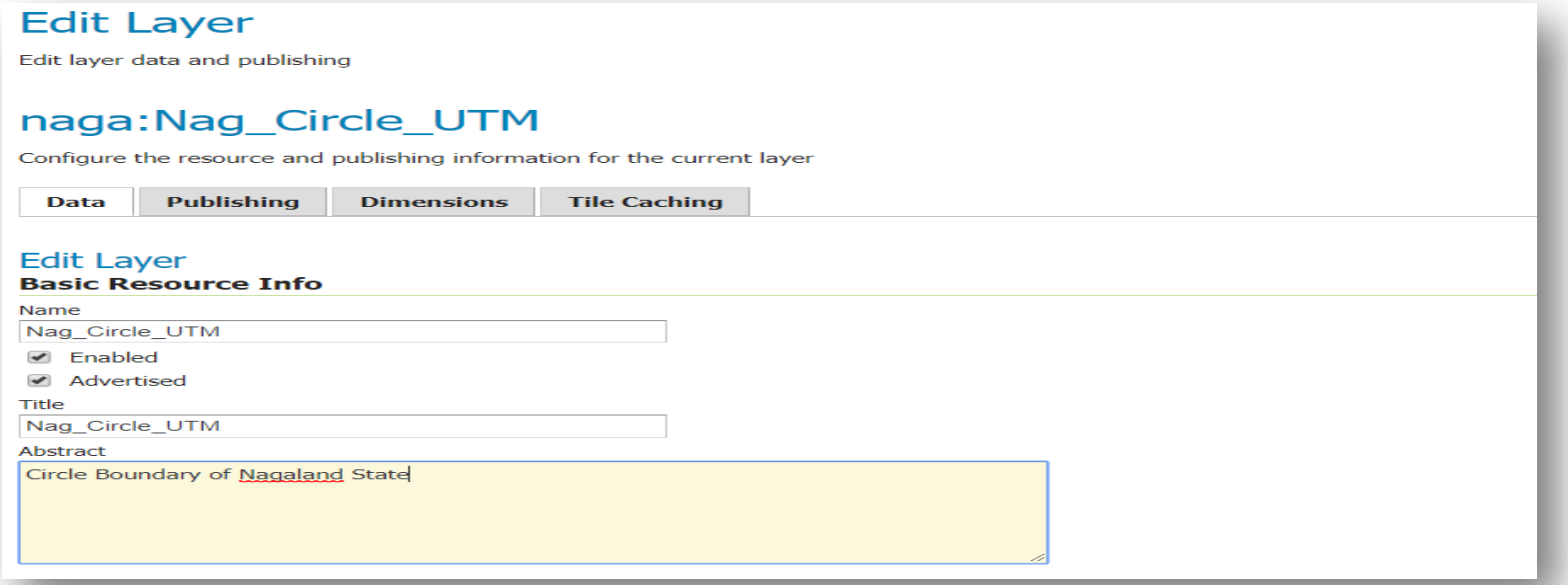
Basic Store Info and Connection Parameters

Publishing Data in GeoServer

1. On the **New Layer** page, click **Publish** beside the **Nag_Circle_UTM** layer name

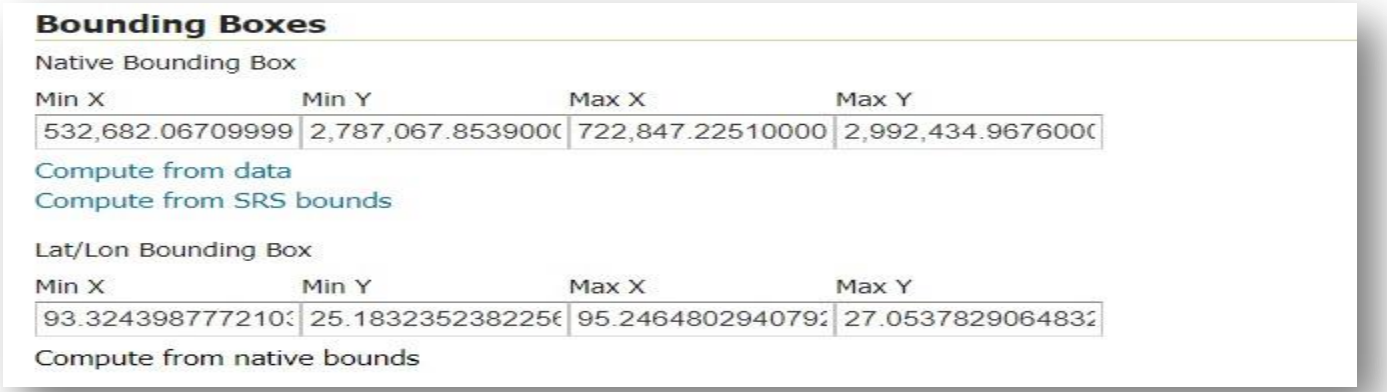


2. The **Edit Layer** page defines the data and publishing parameters for a layer. Enter a short **Title** and an **Abstract** for the Nag_Circle_UTM layer.



Publishing Data in GeoServer

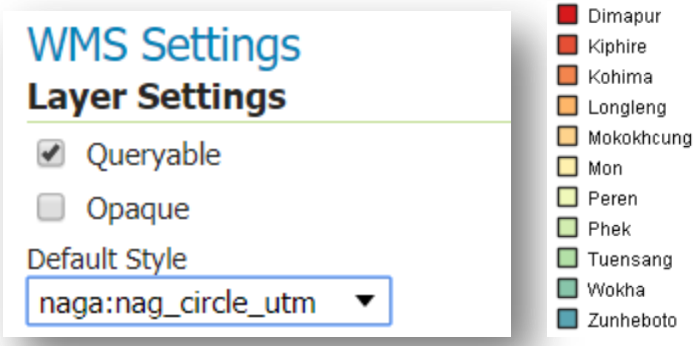
3. Generate the layer‟s bounding boxes by clicking the **Compute from data** and then **Compute from native bounds** links



Generating bounding boxes

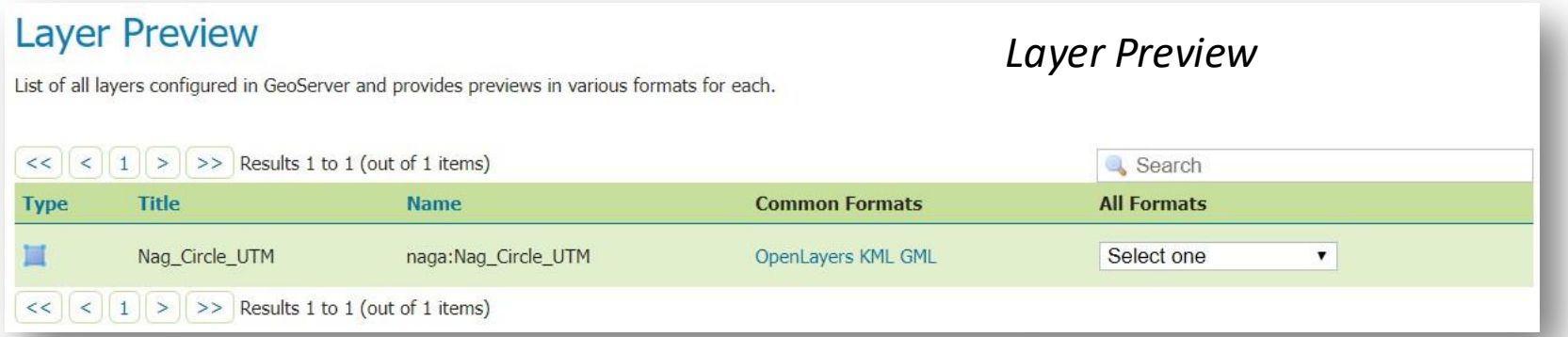
4. Click the **Publishing** tab at the top of the page.

5. We can set the layer‟s style here. Under **WMS Settings**, ensure that the **Default Style** is set to **line**.



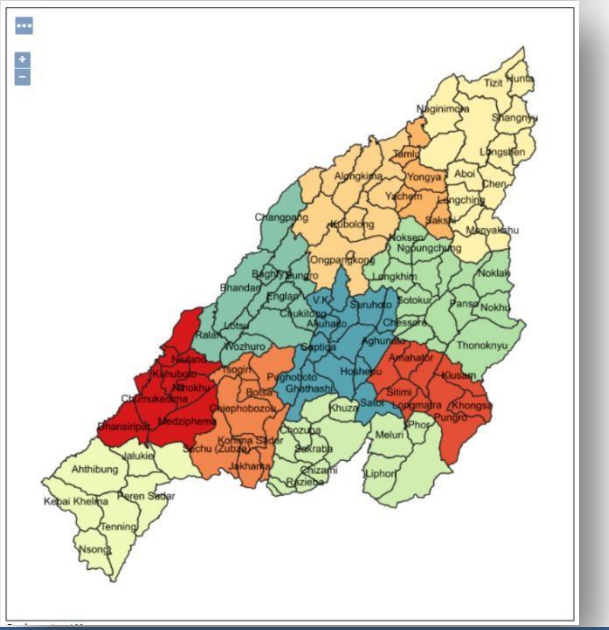
Publishing Data in GeoServer

1. In order to verify that the Nag_Circle_UTM layer is published correctly, we can preview the layer.



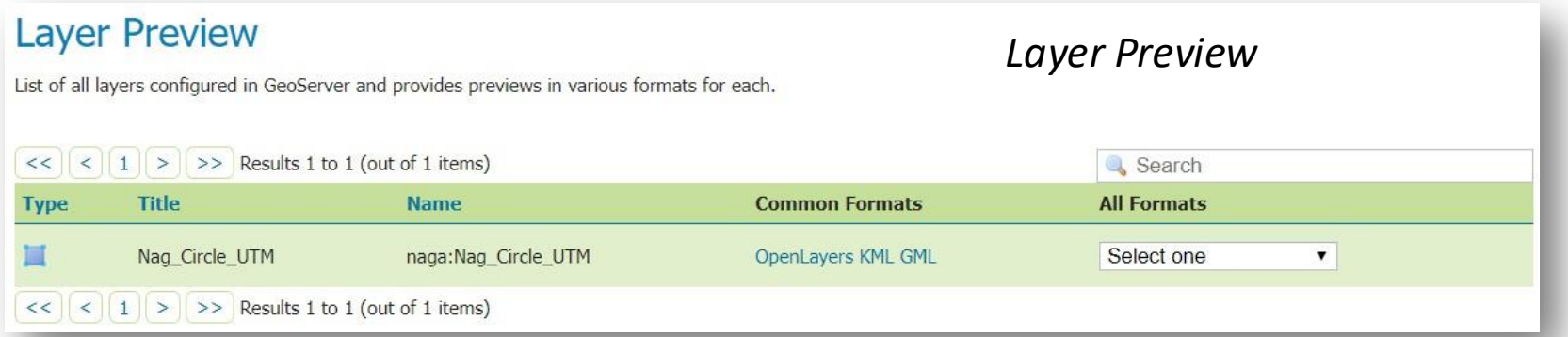
Layer Preview

2. Click the **OpenLayers** link in the **Common Formats** column.
3. An OpenLayers map will load in a new tab and display the shapefile data with our custom style for the layer. You can use this preview map to zoom and pan around the dataset, as well as display the attributes of features.



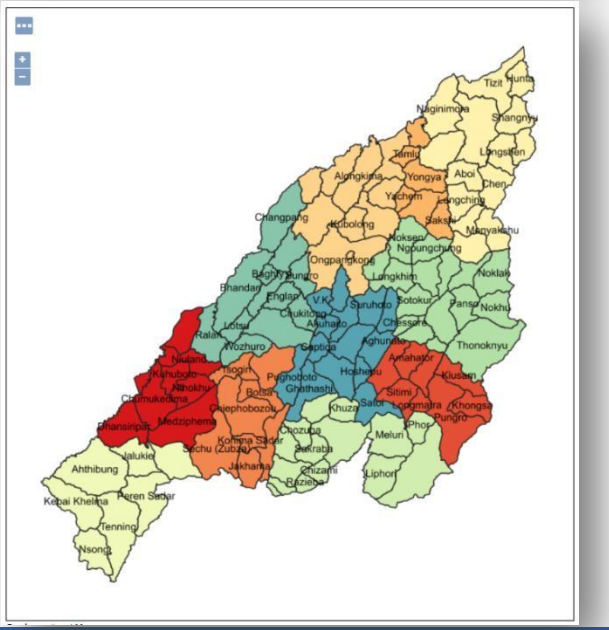
Publishing Data in GeoServer

1. In order to verify that the Nag_Circle_UTM layer is published correctly, we can preview the layer.



Layer Preview

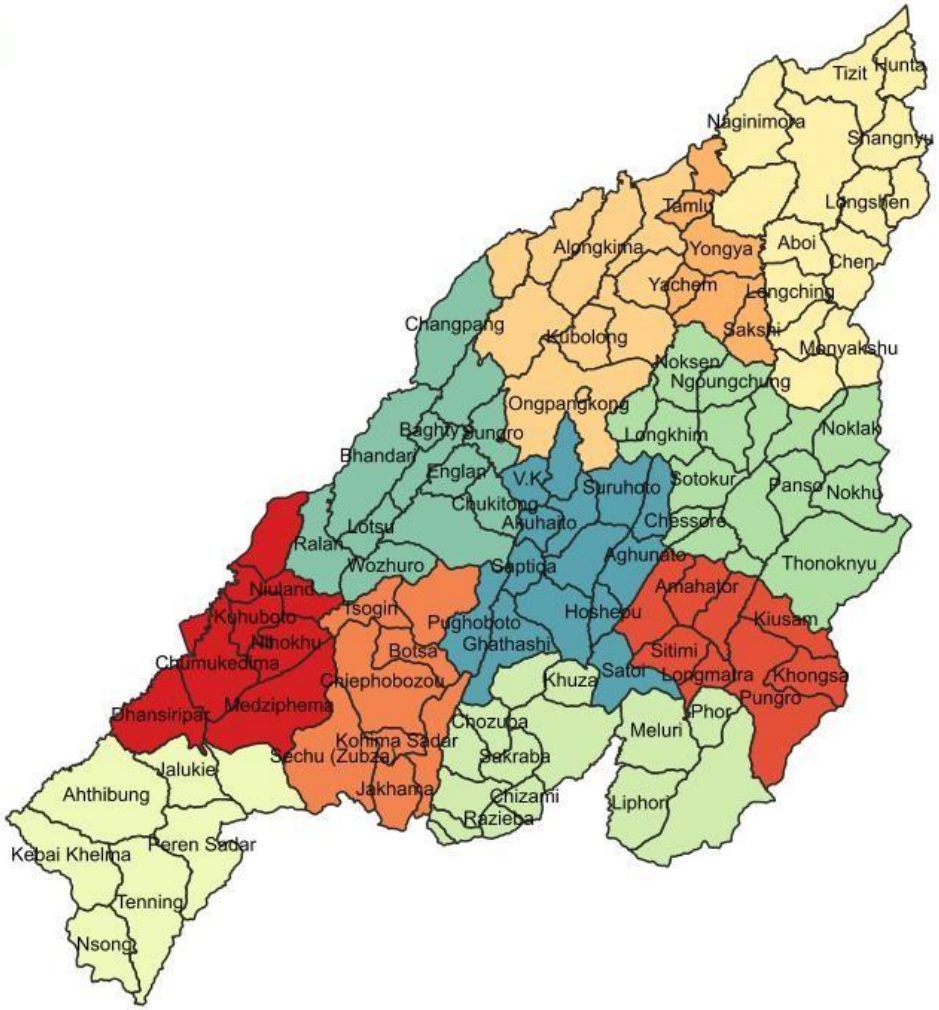
2. Click the **OpenLayers** link in the **Common Formats** column.
3. An OpenLayers map will load in a new tab and display the shapefile data with our custom style for the layer. You can use this preview map to zoom and pan around the dataset, as well as display the attributes of features.



Understanding a WMS Service

The request
The following URL is a WMS 'GetMap' request:

http://localhost:8080/geoserver/naga/wms?service=WMS&version=1.1.0&request=GetMap&layers=naga:Nag_Circle_UTM&styles=&bbox=532682.0670999996,2787067.8539000005,722847.2251000003,2992434.9676000006&width=711&height=768&srs=EPSG:32646&format=image/png&TRANSPARENT=TRUE



Understanding a WMS Service: ‘GetMap’ Request

Breaking down the request

That big, long URL is actually made up of many small bits, separated by '**&**' characters. Here is the request, broken up so that each bit is on its own line, and with bits re-arranged so they flow better:

http://localhost:8080/geoserver/naga/wms?
service=WMS&
version=1.1.0&
request=GetMap&
layers=naga:Nag_Circle_UTM&
styles=&
bbox=532682.0670999996,2787067.8539000005,722847.2251000003,2992434.9676000006&
width=711&
height=768&
srs=EPSG:32646&
format=image/png
TRANSPARENT=TRUE

Understanding a WMS Service: ‘GetMap’ Request

http://localhost:8080/geoserver/naga/wms?
The protocol, host and path

service=WMS&
The *SERVICE* parameter tells the server which exact service you're sending your message to. In some cases, the service endpoint might work for multiple services, and this parameter could be used to specify whether you're sending your 'GetMap' request to the WMS or the WFS.

version=1.1.0&
The 'VERSION' parameter

request=GetMap&
The *REQUEST* parameter tells the server which operation you'd like to perform. The **'GetMap' request tells the server you want to fetch a map image**. Other request types include 'GetLegendImage', 'GetFeatureInfo', 'GetCapabilities, „GetLegendGraphic„

Understanding a WMS Service: 'GetMap' Request

srs=EPSG:32646&
The *SRS* parameter tells the WMS which coordinate system the *BBOX* parameter is expressed in. The SRS parameter is written as an EPSG code.

format=image/png
The *FORMAT* parameter specifies the format of the returned image

TRANSPARENT=TRUE
The *TRANSPARENT* parameter specifies whether areas of the map which are not otherwise drawn should be marked as transparent in the response image

Other Parameters:
http://www.opengeospatial.org/standards/wms

GeoServer CQL

- CQL (Common Query Language) is a **query language created by the OGC** for the Catalogue Web Services specification. CQL is written using a text-based syntax. It is thus more readable and better-suited for manual authoring.

- Common_Query_Language_or_CQL can be a **shorter** (as compared to other filters), more readable way to put a "filter" or SQL-like "where" statement into a URL.

- ECQL is an extension of CQL and thus provides a more flexible language with stronger similarities with SQL.

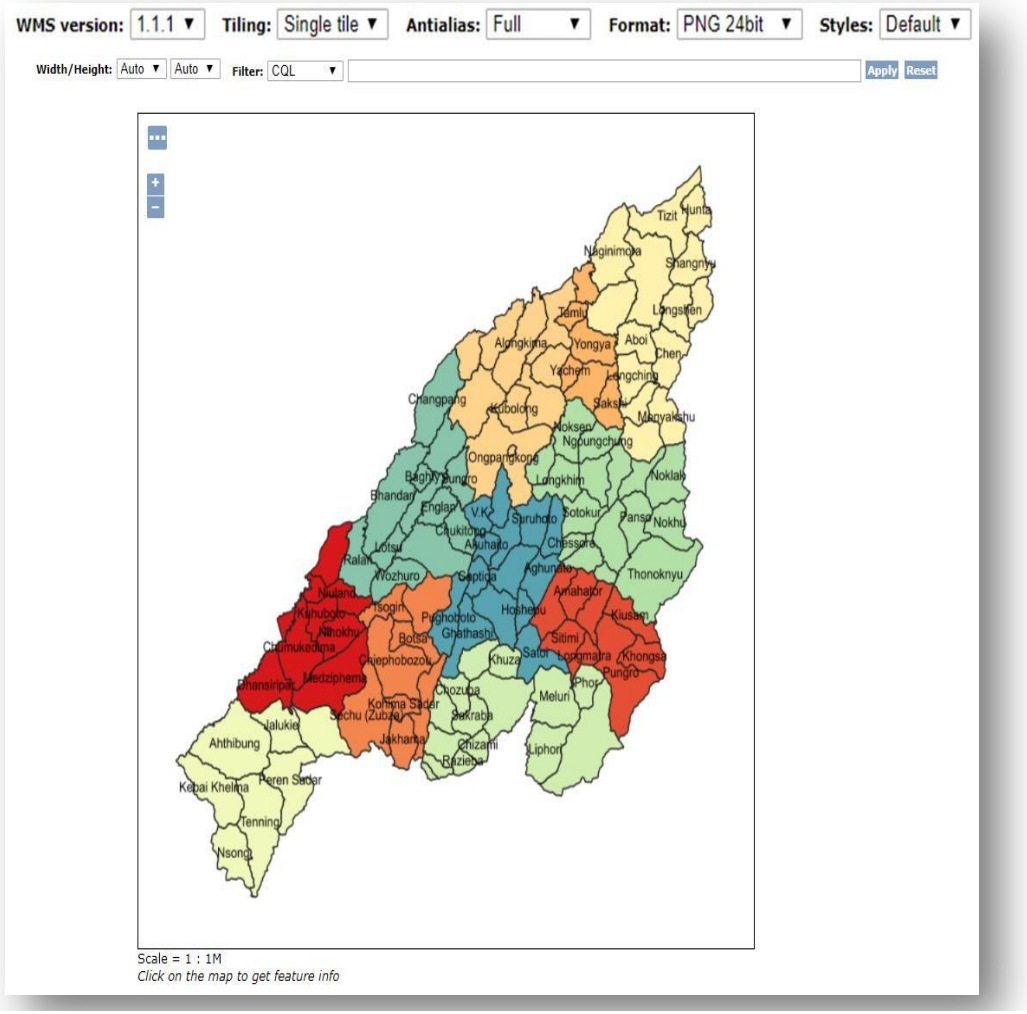
- GeoServer supports the use of both CQL and ECQL in WMS and WFS requests, as well as in GeoServer‟s SLD dynamic symbolizers. Requests can contain attribute statements or spatial requests.

Use **cql_filter** to define filter parameters in your application (OL/Leaflet)

GeoServer CQL

The following examples use the **naga:Circle Boundary of Nagaland** . It demonstrates how CQL filters work by using the WMS [CQL_FILTER vendor parameter] to alter the data displayed by WMS requests.

- Open the GeoServer Map Preview for the **naga:Circle Boundary of Nagaland** layer.
- Click on the *Options* button at the top of the map preview to open the advanced options toolbar.
- The example filters can be entered in the *Filter: CQL* box



Attributes Names

Circle Boundary of Nagaland

fid	OBJECTID_1	OBJECTID	Id	District_N	Circle_N	Shape_Leng	Shape_Le_1	Shape_Area

GeoServer CQL

Layer : Load Nagaland block boundary in Geoserver

Simple comparisons
The full list of comparison operators is: **=, <>, >, >=, <, <=**

CQL query:

```
District_N ='Zunheboto'
Circle_N LIKE 'A%'
Shape_Area > 300
```

list comparisons
If instead we want to extract the states whose name is in a given list we can use the IN operator specifying an attribute name, as in

```
Circle_N IN ('Meluri','Liphori','Lotsu')
```

GeoServer CQL

Filter functions

CQL/ECQL can use any of the filter functions available in GeoServer. This greatly increases the power of CQL expressions.

For example, suppose we want to find all circles whose name contains an “a”, regardless of letter case. We can use the `strToLowerCase` to turn all the circle names to lowercase and then use a like comparison: `strToLowerCase(Circle_N) like '%a%'`

```
strToLowerCase(Circle_N) LIKE '%a%'
```

GeoServer CQL

Geometric filters
CQL provides a full set of geometric filter capabilities. Say, for example, you want to display only the district boundary that intersect the (x,x,x,x) bounding box
Check the current BBOX values from URL and change the bbox and supply it to BBOX query

```
BBOX(the_geom, 532682.0670999996,2787067.8539000005,722847.2251000003,2992434.9676000006)
```

The full list of geometric predicates
is: `EQUALS, DISJOINT, INTERSECTS, TOUCHES, CROSSES, WITHIN, CONTAINS, OVERLAP S, RELATE, DWITHIN, BEYOND`

Eg Using CQL filter for WFS GetFeature as URL.

```
http://host:port/geoserver/wfs?service=WFS&version=1.0&request=GetFeatu
re&typeName=myLayer&CQL_FILTER=INTERSECTS(the_geom,
POLYGON((...)))&propertyName=data1,data2,data3
```

More Ref : https://docs.geoserver.org/2.7.1/user/filter/ecql_reference.html#spatial-predicate

Thank You