Intro to Rasters in Geoserver

Hannah Gedlaman

July 2025

1 Purpose

This document serves as an entry point for learning how to host raster data for efficient visualization and access using GeoServer. It is designed to guide you through basic setup and workflows, supported by linked resources and tutorials that I found helpful during my own learning process. The content builds progressively, following the types of layers and data I have hosted with GeoServer We start with a single raster layer (GeoTIFF), then an image mosaic of GeoTIFFs, and finally NetCDF files and NetCDF-based image mosaics.

While GeoServer is capable of serving both raster and vector data, this guide focuses specifically on raster data. GeoServer is a powerful and flexible tool with many capabilities beyond what is covered here.

I strongly recommend following the linked GeoServer tutorials and using the official documentation as your primary reference. This document is intended to supplement those resources by organizing the key concepts and pitfalls that I encountered while configuring a GeoServer instance to serve large time series of raster data.

2 GeoServer Overview

For a high-level overview of GeoServer, visit: `https://geoserver.org/about/`

Throughout this tutorial we will be accesses our published geoserver data via WMS. to learn specifically about Web Map Services (WMS), see: https://www.ogc.org/standards/wms/

3 GeoServer Download

Install GeoServer for your operating system by following the instructions in the official documentation: `https://docs.geoserver.org/main/en/user/installation/index.html`

Additional (and for this tutorial - necessary) functionality is provided via plugins. These extensions must match the exact version of your core GeoServer installation. You can find them on the official GeoServer download page under the "Extensions" section.

Installing Plugins

To install a plugin:

1. Download the desired extension as a '.zip' file from the GeoServer version-matched extensions page.
2. Extract the contents of the '.zip'.
3. Copy all extracted '.jar' files into the 'WEB-INF/lib' directory of your GeoServer installation.
    - For a standard installation, this is typically located in: `$GEOSERVER_HOME/webapps/geoserver/WEB-`
4. Restart GeoServer for the plugin to be loaded.

Recommended Plugins

For hosting NetCDF image mosaic data (for completing all steps detailed in this tutorial), You should install the following extensions:

- **Image Mosaic**
- **NetCDF**
- **NetCDF Output Format**

Next Steps

At this point, it's a good idea to explore the GeoServer interface and get familiar with the layout. You may also want to try one or more of the official Getting Started tutorials: `https://docs.geoserver.org/stable/en/user/tutorials/index.html`

4 Publishing a Raster Layer

To get started, I recommend publishing a basic raster layer by following this tutorial: `https://docs.geoserver.org/latest/en/user/data/raster/geotiff.html`

Here is a sample geotif file that you can donwload and use alongside the above linked tutorial to publish your first raster layer: `https://github.com/hged13/cheatsheet/blob/main/median.tif`

The general process for publishing any layer works as follows:

1. First, publish your raster file (e.g., a GeoTIFF) as a **raster store**.
2. Once the store is saved, you will be prompted to publish it as a **layer**.

Assigning a Style (SLD)

In order to visualize the published layer, you must assign it a **Styled Layer Descriptor (SLD)** that maps values to colors. This step is essential. Without a style, your layer will not appear correctly in previews or maps. You can assign a style while publishing, or while editing a published layer.

The SLD should reflect the relevant value ranges of your data and assign meaningful color mappings. You can find example SLDs and styling approaches for rasters in the GeoServer SLD Cookbook: `https://docs.geoserver.org/main/en/user/styling/sld/cookbook/rasters.html`

Creating and Publishing a Custom SLD

If you are using the sample data I have provided, you can publish the following sld file as a style : `https://github.com/hged13/cheatsheet/blob/main/style.sld`

To use a custom SLD:

- You must first **publish** the SLD in GeoServer. This can be done by following the official tutorial here: `https://docs.geoserver.org/latest/en/user/gettingstarted/style-quickstart/index.html`
- A quick way to generate an SLD file is by styling your raster data in QGIS, then exporting the style: `https://docs.geoserver.org/main/en/user/styling/qgis/index.html`
- After the style is uploaded, you must assign it as the **default style** for your layer in the GeoServer layer `Publishing` tab. (this can be accessed when configuring or editing a layer)

Previewing and Viewing the Raster

After you have published a style, and followed the raster publishing tutorial you should be able to view your raster:

- Navigate to the `Layer Preview` section in the GeoServer web interface.
- Select your published layer and choose a preview format (e.g., OpenLayers).

Visualizing in a Web Map (Leaflet)

To view your published raster data in a simple web map using Leaflet, you can use the provided HTML viewer template, available at:
`https://github.com/hged13/cheatsheet/blob/main/view.html`

Be sure to update the layername and geoserver url variables to reflect your spcecific geoserver and layer information.

This viewer loads your GeoServer-published raster layer using the Web Map Service (WMS) protocol and displays it in a standalone browser-based map. Be sure to update the GeoServer URL and the layer name in the HTML file to match your geoserver instance and layer name respectively.

This basic viewer can be reused for any layer you publish throughout this tutorial. Just make sure you modify the `layer` parameter in the view file accordingly.

Launching the Viewer Locally To open the viewer in your browser:

1. Save the HTML file (e.g., `viewer.html`) in a local directory.
2. From that directory, start a simple local web server by running the following command in your terminal:

   ```
   python3 -m http.server 8000
   ```

3. Open your browser and navigate to:

   ```
   http://localhost:8000/viewer.html
   ```

The viewer should load your WMS layer into an interactive Leaflet map.

5 Requesting data from Geoserver

The Leaflet visualization example works by sending a WMS (Web Map Service) request to GeoServer. Leaflet is designed to integrate seamlessly with GeoServer, but what is actually happening behind the scenes is that we are requesting the raster data that we previously uploaded. This approach also supports time series data, allowing us to request a specific time or range of times, as well as particular bounding boxes.

The requested data can be delivered in various formats. For example, you can paste a WMS request URL directly into a browser and view the corresponding image of your layer. For guidance on formatting GeoServer WMS requests,

see here:
`https://docs.geoserver.org/2.21.x/en/user/services/wms/reference.html`.

GeoServer also supports other services beyond WMS. You can find an overview of available services here:
`https://docs.geoserver.org/latest/en/user/services/index.html`.

For raster data retrieval, WCS (Web Coverage Service) is particularly relevant. A WCS request allows you to retrieve the raw data itself, rather than just an image. WCS resources can be found here: `https://docs.geoserver.org/latest/en/user/services/wcs/index.html`.

6 Image Mosaics

Image mosaics in GeoServer let you combine multiple raster files into a single layer. This is especially useful when you have files that represent different spaces, times, or elevations. The typical setup uses GeoTIFF files, each of which contains a single band of data. A setup also requires a way to index the datat. Geoserver's default is to use a shapefile, but it is better practice to use a different method. For this tutorial we will be using a postgis database.

You can read about how to publish an image mosaic here:
`https://docs.geoserver.org/latest/en/user/data/raster/imagemosaic/configuration.html`

In the following few sections I will provide some sample data and walk you through setting up a postgis database. This, in conjunction with the linked tutorial above, should allow you to publish your first image mosaic.

Pay close attention to the requirements for the configuration files. (which are described in the above tutorial.) `indexer.properties`, defines how GeoServer indexes your mosaic data. Ensure that your filename structure matches the expected pattern defined in the `indexer.properties` and `time.regex.properties` files, so GeoServer can correctly parse and index your files. If these are not setup correctly, you will be unable to publish your layer with time support. You can reference the filename structure and configuration files in my linked sample data directory for a reference of how this setup might look.

You can download this directory with 5 tif files, along with datastore.properties, indexer.properties, and time.regex.properties files here:
`https://github.com/hged13/cheatsheet/tree/main/image_mosaic`

Make sure all the files are placed in the same (otherwise empty) directory after you have downloaded them, and read the next section to learn how to correctly setup the postgis database before trying to publish the mosaic.

7 PostGIS and Image Mosaic Configuration

By default, GeoServer indexes image mosaics using a shapefile-based index. However, it is generally recommended to use a different indexing method if you are working with more data. In this guide, I will detail the process for using a PostGIS-enabled PostgreSQL database as the indexing backend. You only need to provide geoserver with an empty schema, and it will do all of the necessary population work for you.

For the following steps, it is assumed that you have access to the PostgreSQL command-line tool (psql), pgAdmin, or another PostgreSQL interface. The following commands should be executed from within one of these tools.

Overview

To configure a PostGIS-backed for an image mosaic:

1. Ensure you have access to a PostgreSQL database with the PostGIS extension enabled.

In order to successfully publish this mosaic, you must configure a PostgreSQL database, schema, and user that match the parameters defined in your `datastore.properties` file.

The following setup corresponds to the example configuration. Any change in names should be reflected in the datastore.properties file as well:

1. Create the Database

If a PostgreSQL database named `geoserver_db` does not already exist, create it:

```
CREATE DATABASE geoserver_db;
```

2. Create the User

Create a PostgreSQL user named `geoserver_user` with an appropriate password:

```
CREATE USER geoserver_user WITH PASSWORD 'password';
```

3. Create the Schema

Within the `geoserver_db` database, create a schema named `mosaic_schema`:

```
\c geoserver_db
```

```
CREATE SCHEMA mosaic_schema AUTHORIZATION geoserver_user;
```

While connected to the db, enable PostGIS:

```
CREATE EXTENSION postgis;
```

4. Grant Permissions

Ensure the user has the necessary privileges on the schema:

```
GRANT ALL PRIVILEGES ON SCHEMA mosaic_schema TO geoserver_user;
```

5. Define Connection Parameters

In the mosaic directory, include a `datastore.properties` file with the following content: (if you downloaded the sample direcotry, this should already be included)

```
SPI=org.geotools.data.postgis.PostgisNGDataStoreFactory
database=geoserver_db
user=geoserver_user
passwd=password
host=localhost
port=5432
schema=mosaic_schema
```

After completing these steps, the mosaic should be ready for publishing through GeoServer. Walk through the necessary steps that are detailed here: `https://docs.geoserver.org/latest/en/user/data/raster/imagemosaic/configuration.html`

While publishing the layer you need to go to the **Dimensions** tab and enable the **time** and/or **elevation** dimensions if your mosaic includes them. If these options are not available to enable, it means GeoServer did not detect those dimensions from your mosaic, indicating a likely misconfiguration. In that case, you should check the geoserver logs, review your setup carefully or consider restarting the tutorial to ensure the dimensions and naming conventions are correctly defined in your configuration files.

You should also assign the same SLD you used for the raster layer publishing section above in order to properly view the layer. Then, you should be able to view a layer preview, or view your layer on the leaflet map by changing the layer name as described in the above section. If you don't already have the sld downloaded, you can find it here:
`https://github.com/hged13/cheatsheet/blob/main/style.sld`

Note: If you attempt to re-publish or update a mosaic using the same postgres database schema (even if publishing failed and you are simply trying again)

, you must clear any tables that geoserver created within your schema. Otherwise, GeoServer may encounter errors or fail to refresh the mosaic correctly. The simplest approach is to drop and recreate the schema before reusing it.

If you need to publish data with multiple bands or include many timesteps within a single file for easier access, using the NetCDF format and/or a NetCDF mosaic might be a better option. This will be explained in the following sections.

8 Publishing NetCDF Layers in GeoServer

If you want to publish a layer in GeoServer with similar functionality to an image mosaic, but without using a directory of single-band files, you can publish a NetCDF raster store. Detailed instructions can be found in the official documentation:
`https://docs.geoserver.org/main/en/user/extensions/netcdf/netcdf.html`

To enable this functionality, you must download and install the NetCDF plugin, as mentioned at the beginning of this document. If you missed this, scroll up to the 'Installing Plugins' directions.

To publish a NetCDF file in GeoServer, the file's metadata must conform to the CF (Climate and Forecast) metadata conventions. A sample NetCDF file containing several variables and a time dimension, which is compatible with GeoServer, is provided here: `https://github.com/hged13/cheatsheet/blob/main/fuel_mositure.nc`.

You can inspect the metadata of the file using the following terminal command:

```
ncdump -h {path-to-cdf-file}
```

Additionally, here is an SLD (Styled Layer Descriptor) file that works with the provided NetCDF file (this is the same sld as has been provided prior): `https://github.com/hged13/cheatsheet/blob/main/style.sld`.

After downloading the files, you should be able to publish the NetCDF file in GeoServer similarly to how you publish a GeoTIFF. Simply add a NetCDF data store(This option should appear under raster data sources if the NetCDF plugin is correctly installed) and configure the connection parameters by selecting the file location of your cdf file. Then, publish the associated layers when prompted.

9 File Size Considerations & NetCDF Image Mosaics

When serving large datasets, GeoServer can become unresponsive or fail to load layers if the files are too large. This might happen with netcdf files that contain a lot of data. To improve performance and stability, consider splitting your dataset into smaller, more manageable NetCDF files and publishing them as a mosaic.

For example, I found success by dividing several years of hourly data into individual NetCDF files, each covering 48 hours (two days). These files were approximately 5–6 GB in size and performed reliably in GeoServer.

To create a NetCDF image mosaic, refer to the **Mosaic of NetCDF Files** section in the GeoServer documentation: `https://docs.geoserver.org/main/en/user/extensions/netcdf/netcdf.html`

Unlike GeoTIFF mosaics, the NetCDF extension includes a command-line utility that generates the required configuration files (e.g., `indexer.xml`, `datastore.properties`). But this means that the netcdf files included in the mosaic must be structured as expected.

Important: All NetCDF files included in the mosaic must:

- Conform to CF (Climate and Forecast) metadata conventions
- Share identical structure—including variable names, dimensions and coordinate systems

If these conditions are not met, GeoServer may fail to detect dimensions such as `time` or `elevation`, and the mosaic will not load or function as expected. I've linked a working netcdf image mosaic directory you can upload to geoserver, and use to understand the structure of netcdf image mosaics.
`https://github.com/hged13/geoserver_cheatsheet/tree/main/netcdf_mosaic`

After downloading the example files, and placing them in the same (otherwise empty) directory, you should be able to run the command-line utility from within that directory and generate all necessary configuration files.

As covered in the Image Mosaic section, it is necessary to configure an empty database schema that matches the settings defined in a datastore.properties file, which you must place in the appropriate directory. The sample directory should already include a datastore.properties file. Follow the database configuration steps from the Image Mosaic section, ensuring that the schema name, username, password, and other details in datastore.properties match the database you are creating.

If the schema already exists (for example, if you created it when publishing the initial image mosaic), you have two options: either update the schema name in datastore.properties to use a new schema, or drop and recreate the existing schema in PostgreSQL.

After these steps have been taken, you can publish the directory as an image mosaic. It may take some time to load, especially if the mosaic contains many large files.

10 Using a Custom CRS in GeoServer

GeoServer allows you to define and register custom Coordinate Reference Systems (CRS), which is useful when working with local projections or non-standard systems.

Follow the official guide to add a custom CRS: `https://docs.geoserver.org/latest/en/user/configuration/crshandling/customcrs.html#crs-custom`

To summarize the steps:

1. Create or modify the `user_projection_defs.properties` file located in:
   - `$GEOSERVER_DATA_DIR/user_projections/`
2. Add your custom CRS definition using a unique EPSG code (e.g., `EPSG:9999`) and the full PROJ string or WKT representation.
3. Restart GeoServer to load the new CRS definition.

Important: If you're using a custom CRS with an **image mosaic**, there are two places where the CRS must be registered:

1. It must be defined and recognized by GeoServer, using the method described above.
2. It must also be included in the PostGIS index used by your mosaic (if applicable). This requires the custom CRS to be defined within the PostGIS database itself.

In PostGIS, you can add a custom CRS using SQL. For example:

```
INSERT INTO spatial_ref_sys (srid, auth_name, auth_srid, proj4text, srtext)
VALUES (
  9999,
  'EPSG',
  9999,
  '+proj=... (your proj string here)',
  'GEOGCS["...", DATUM[...]]'
);
```

You can verify whether GeoServer correctly recognizes the custom CRS by navigating to the `Coordinate Reference Systems` section in the Admin interface and searching for your EPSG code.