EXERCISE: Publishing spatial data with GeoServer

Barend Köbben
Ivana Ivánová
August 30, 2015

Contents

1 Introduction .................................................. 2
2 GeoServer’s main concepts .................................. 2
3 Publishing spatial dataset to the GeoServer .......... 5
   3.1 Creating a workspace ........................................ 6
   3.2 Connecting to an existing data source — adding a store .......... 7
   3.3 Publishing a layer .............................................. 8
   3.4 Displaying all layers in one WMS ......................... 10
4 Summary .................................................. 11
Objectives of this exercise

In this exercise you will learn the how to:

1. define a workspace on GeoServer,
2. connect to an existing data repository,
3. expose spatial dataset as an OGC compliant service, and
4. create a WMS serving a group of layers to the user.
1 Introduction

GeoServer\(^1\), an open source server for sharing spatial data, is an application, which was built on the vision of geospatial web, a web where people not only share information in documents, but also exchange spatial data. With GeoServer, spatial data is served to the users through interoperable interfaces standardized by the Open Geospatial Consortium (OGC)\(^2\):

- Web Map Service (WMS) — for serving vector spatial data in a form of maps in an image or document formats (e.g. .png, .gif, .svg, .pdf)
- Web Feature Service (WFS) — for serving vector spatial data in various formats (e.g. .gml, GeoJSON, .kml, .csv, .shp), and
- Web Coverage Service (WCS) — for serving raster spatial data in an image or document formats (e.g. .png, .gif, .svg, .pdf)

GeoServer also supports OGC’s style definition standard (SLD), which allows the use of predefined symbology and style for a layer.

In this exercise we will learn how to publish a spatial dataset to GeoServer and how to expose this published layer as a web map service. However, before starting the administration, it is necessary to introduce the main concepts used in GeoServer.

2 GeoServer’s main concepts

As every application, GeoServer uses its known terminology. Here is the explanation of the terms we will be using frequently from now on:

**Workspace**: is a container for organizing items. It is analogous to the namespace in the XML language. In GeoServer, the workspace is used to group items (e.g. stores, layers) that belong to the same universe. In our case, every user on the GeoServer will have his or her own workspace. Workspaces allow having different layers with the same name, each of them assigned to their own workspace. For example, for storing census data for various states, we could have a workspace for each state and same type of data (called by the same name) assigned to the state workspace where they belong — SP:distritos, RJ:distritos.

**Store**: connects to a data source, which can be file (e.g. a .shp), directory of files, or a database containing vector or raster data. It is necessary to define a store before publishing any data as a layer. Store offers an advantage of one time connection definition (e.g. to a PostgreSQL/PostGIS database) for publishing a several datasets from the same data source.

**Layer**: refers to a spatial dataset (vector or raster) that contains geographic features. Layers represent each feature that needs to be shown on a map. Each layer comes from an existing Store.

---

\(^1\)http://geoserver.org/
\(^2\)http://www.opengeospatial.org/
Layer Group: a collection of layers (organized hierarchically) allowing request of multiple layers with a single WMS GetMap request. An example that comes as part of the GeoServer installation is a Layer Group called tasmania. WMS request to this layer group is:


And the result of this request is:

Observe, that the layer group tasmania, contains three layers: tasmania_state_boundaries, tasmania_water_bodies, and tasmania_roads. However, observe that in the WMS request we are asking only for the layer group (... &layers=tasmania&...). Without the concept of the layer group, to see the map of Tasmania, users would need to request all three layers: ... &layers=tasmania_state_boundaries,
tasmania_water_bodies,tasmania_roads... Layer Group concept has advantages — when we want to publish individual layers that constitute a map and want our user to see them ‘assembled’ the way they should be, we would create a Layer Group and advertise it in the capabilities document as one layer only.

Style are written in Styled Layer Descriptor (SLD), an OGC standard we talked about in the previous exercise. By defining styles for a layer, we can enforce the way it is rendered to the user — put simply, if a cartographic standard says, a river should be portrayed as a 1 pixel wide blue line, using SLD language we can tell GeoServer to always serve river layer as a 1 pixel wide blue line:

```xml
...<FeatureTypeStyle>
  <Rule>
    <Name>rule1</Name>
    <Title>Blue Line</Title>
    <Abstract>A solid blue line of 1 pixel width</Abstract>
    <LineSymbolizer>
      <Stroke>
        <CssParameter name="stroke">#0000FF</CssParameter>
      </Stroke>
    </LineSymbolizer>
  </Rule>
</FeatureTypeStyle>
...```
3 Publishing spatial dataset to the GeoServer

After learning the main concepts, we can begin with the GeoServer’s administration. We want to make sure that every GeoServer administrator (i.e. every student in our course) will work in his or her own workspace, publish his or her own datasets and administer his or her own layers. We will ensure this by creating own the workspace, connecting with own parameters to the datastore and publishing a layer, assigning own style to a layer, and, eventually, creating own layer group(s).

In this exercise we will work at a local installation of the GeoServer.

---

**Task 1**: Connect to the GeoServer with following connection parameters:


If you are working on a desktop computer in the lab:

username: admin

password: geoserver

If you are working on your own laptop use the administrator login credentials you set during the installation of the GeoServer.

Look around and familiarize yourself with GeoServer’s web administration interface — here is a quick overview scheme to help you:

![GeoServer Overview Scheme](image)

In this overview scheme, we have GeoServer’s main administration menu with sections providing access to:
1. overview: overview of the published content

2. server info&settings: access to configuration and diagnostic tools

3. data: administration of the spatial content:
   - Layer preview: overview of all published layers with option to preview these layers in several supported formats
   - Workspaces: management of workspaces
   - Stores: connection to spatial data stores
   - Layers: layer management: adding/removing and editing layers
   - Styles: style definition management

4. services: settings management of WCS, WFS and WMS services

5. tile caching: settings of predefined scale-dependent map tiles

6. security: security settings (e.g. users and administrators authorization)

7. demos&tools examples, demonstrations and tools enhancing the use and functionality of the GeoServer

8. server capabilities: overview of OGC standard interfaces and their versions supported by the GeoServer

Let us now begin the server administration with the ultimate aim to publish our a spatial dataset to the GeoServer as Web Map Service

3.1 Creating a workspace

As we explained earlier, a workspace is a useful construct that helps us contextualize the content on the server. In our context, every server administrator will work in his or her own workspace. In practice, the context is not always defined by server administrators (as this group is usually not big), it may also be defined by the project or a department (as seems to be the case at IBGE’s GeoServer).

**Task 2:** From the Data section select **Workspaces**.

Define your own workspace — select **Add new Workspace** and specify its:

- **Name** — On the GeoServer, the workspace **Name** is limited to 10 characters and may not contain space. Let’s agree that in our exercise a workspace name will be your own name.
• **Namespace URI** — A URI is similar to a URL, except URIs don’t need to point to a location on the web; they only need to be a unique identifier. In our exercises, a workspace URI will be associated with a user, where a different workspace name will represent a different trailing identifier. In the case of workspace called `yourname` and the GeoServer is on the localhost (i.e. installed on your working station) this would be `http://localhost:8080/geoserver/yourname`, where `http://localhost:8080/geoserver` represents the URL of UNESP/FCT’s GeoServer and `yourname` is the trailing identifier pointing to the `yourname` workspace on the server.

Remember (or store) this URI, you will need it later in the exercise.

To avoid the looooooooooong list of all supported spatial reference systems, which comes by default by the GeoServer in the capabilities document, we need to specify, which spatial reference systems will be supported by the web services in the newly defined workspace. We can do this by specifying the WMS service capabilities in the workspace.

Open your workspace for editing — on the first page, feel free to add your personal information, after all, you are the owner of this workspace... Next, select from the list of supported Services the WMS service. A WMS service definition window you can edit the service metadata (e.g. to inform your users what can they expect in web map services coming from this workspace), edit accessibility (e.g. fees or other access constraints) and also, you can limit the list of supported spatial reference systems. Our layer is in WGS84 reference system, but we also want to serve it in one other spatial reference systems (SRS), namely in the WGS84 Pseudo-Mercator SRS (this is the system used by e.g. OpenStreetMap). the EPSG codes of these systems are:

- 4326 — WGS84
- 3857 — Pseudo-Mercator

Add the list of supported codes (separated by commas) in to the Limited SRS list and check **Output bounding box for every supported CRS** (this will advertise the spatial extent in every defined SRS for every published layer in your workspaces WMS). Submit your workspace definition.

As a result, you should be able to see your newly created workspace on GeoServer’s list of workspaces.

### 3.2 Connecting to an existing data source — adding a store

We will now publish the layer called `forest`, which we prepared in the previous exercise. As you may recall, this layer ‘lives’ in our `thailand` on the PostgreSQL/PostGIS server
at UNESP/FCT. In GeoServer, we need to connect to this server and we do that by establishing a new store.

**Task 3**: From the Data section select Stores and Add new Store. For adding a new store, we need to specify a kind of data source we will work with — our dataset is a vector spatial dataset and, as we said, it is located in thailand database of the PostgreSQL/PostGIS Database management system hosted at UNESP/FCT server. The connection parameters to this database:

**Workspace**: for assigning a workspace to a store, select from the drop-down menu your own workspace.

**Data Source Name**: give a name to your data source (it can, but it does not have to be called the same as the name of the database)

- **host**: this is the URL of the host of PostgreSQL/PostGIS server — 11d.fct.unesp.br
- **port**: 45432
- **database**: thailand
- **schema**: public
- **user**: webmapUser
- **password**: webmap2015

You may leave other fields with default values and settings and Save the connection settings.

---

### 3.3 Publishing a layer

After adding our database connection to the list of GeoServer’s stores, we now can publish any spatial dataset (i.e. table in a spatial database) as a layer to the GeoServer.

**Task 4**: From the Data section select Layers and Add new resource. The server prompts you to select from existing data stores — select your own data store. You should receive a list of available datasets in the store: find forest and hit the Publish button.

Some layer settings are loaded directly from the dataset, but there are some settings we still need to edit.

In the layer editing menu the first tab is the Data tab. You may add here some metadata (such as title, abstract and keywords describing the dataset), or, if you would have an existing metadata record, you may even add a URL to this resource. We do not have an external metadata resource, hence the only metadata resource for
our user will be the Capabilities document of the WMS we are currently creating. Therefore, you should now edit your layer by adding much metadata as possible.

**Task 5**: Get back to the Title, Abstract and Keywords, and provide an indicative title and provide description of your layer.

We need to tell GeoServer what is the spatial reference system of our data. This information is used for computing the spatial extent of the dataset and for reprojecting the data during both WMS and WFS requests. When loading an OGC compliant spatial datasets (e.g. tables from the PostgreSQL/PostGIS database), the native spatial reference system is loaded directly from the data with the Publish button (that explains why the value **EPSG:4326** is already present in the Native SRS field).

**Task 6**: We now need to tell GeoServer how it should handle the defined spatial reference systems. As we defined only the native SRS for our layer, we need to specify the SRS handling field as the following: **Keep native**. To complete the coordinate reference system settings of our layer, we want to specify the spatial extent (i.e. the bounding box) of the layer: for the Native Bounding Box: **Compute from data** and for the Lat/Long Bounding Box, **Compute from native bounds**.

These are all of the settings we will specify at this moment and we leave the default values in all other fields. Finally we **Save** our settings. As a result, our layer is now published on the server. You may verify this with writing a proper URL GetCapabilities request to your workspace and, from the information learned in the capabilities document, you can construct a GetMap request to ask for WMS of your layer:

**Task 7**: With your service URL ask for the capabilities document of part of the GeoServer’s content (i.e. on your workspace).

Did you forget how to construct a URL request? Okay, for just this once, here is a help:

<your Workspace URI>/wms?service=WMS&request=GetCapabilities

And after careful analysis of the result, you may able able to compose a request like this:

<your Workspace URI>/wms?service=WMS&version=1.1.0&request=GetMap& layers=<your workspace name:layer name>&styles=&bbox=1.083178766365399E7,617871.2532724562,1.1764297326924933E7,2337561.548066317&
Did you find this too complicated? You can ask GeoServer’s to do this for you in the future:

**Task 8**: In the Data section, select the Layer Preview. On the list of layers, find your own layer and instruct GeoServer to show it to you. How will GeoServer do this? It is able to issue GetMap requests asking an WMS in any available output format.

And for creating a complete map of Thailand, we need to publish the remaining three layers.

**Task 9**: Following the steps explained above, publish layers airports, waterbodies and railroad to the GeoServer.

### 3.4 Displaying all layers in one WMS

GeoServer allows creation of layer groups. As we explained earlier, this is a handy construct for situation if you want your user accessing a pre-defined group of layers. Our example is just like that - we would like to allow users of our simple Thailand map accessing the content as one WMS. Let’s group the layers in to a map.

**Task 10**: From the Data section, select Layer Groups and then Add new layer group. Define a name for the group — a sensible name in our case would be ‘thailand’. Next, you can define title for your map and write a short text (in Abstract) explaining users of your WMS what your ‘thailand’ map contains.

Next steps are similar to the steps during publishing a layer — you need to define a workspace (your workspace) to which the layer group will belong, the spatial reference system (EPSG:4326) in which the layer group will be defined and the bounding box representing the spatial extent of the map (lat & long extent for Thailand: Min X = 97.35, Min Y = 5.61, Max X = 105.65, Max Y = 20.47).

As this is a layer group, obviously, we need to define which layers belong to the group — select the layers (with Add layer) and define drawing order (using up/down arrows) for rendering these layers.

You can leave the rest with default settings and save your layer group definition.

After this, your users are able to issue requests to a group of your services with request like this: 
http://localhost:8080/geoserver/webmap/wms?service=WMS&version=1.1.0&
request=GetMap&layers=webmap:thailand&styles=&bbox=97.35,5.61,105.65,20.47&
width=428&height=768&srs=EPSG:4326&format=application/openlayers

We reached the end of this exercise and as a result, you now have your own working
Web Map Services. To verify if users will be able to use it, try:

- requesting capabilities of this service,
- based on the information in the capabilities, request an map (image or a docu-
  ment) with this layer, and
- try connecting to your service from QGIS.

4 Summary

In this exercise we practiced GeoServer administration and you should now be able to:

- work with workspaces,
- connect to the existing data source,
- publish layer as web map service, and
- create a WMS serving a group of layers to the user.