Geomajas vendor specific pipeline plug-in guide

Geomajas Developers and Geosparc

Geomajas vendor specific pipeline plug-in guide

by Geomajas Developers and Geosparc

1.16.5

Table of Contents

List of Examples

Chapter 1. Introduction

The vendor specific pipeline plugin provides custom pipeline definitions that work around some known limitations of vendor-specific data sources. We currently know of the following limitations:

- ESRI WFS servers have no support for passing a composite spatial filter consisting of multiple spatial subfilters within a WFS GetFeatures request
- Oracle spatial database shows poor performance when passing multiple spatial restrictions in an SQL query. It seems that the Oracle database is not as good as PostGIS in using the spatial index in an optimal way by e.g. reordering filters.

The workaround for both cases turns out to be the same. The Geomajas pipeline applies both a "tile bounds" filter and an "allowed area" filter to the layer. By splitting this up in a two-stage process and only applying the first stage when querying the layer, the spatial filter is reduced to a single "tile bounds" filter. The "allowed area" or security filter has to be applied in-memory in a second stage, of course.

This plugin has been successfully tested on a ESRI WFS server and Oracle database.

Chapter 2. Configuration

The configuration is limited to adding the dependency and configuring the correct pipeline where needed.

1. Dependencies

Make sure your include the plug-in in your project. If you are using Maven, add the following dependency to your pom:

Example 2.1. Plug-in dependency

```
<dependency>
   <groupId>org.geomajas.plugin</groupId>
   <artifactId>geomajas-plugin-vendorspecificpipeline</artifactId>
    <version>1.16.5</version>
  </dependency>
```

2. Pipeline configuration

To override the default pipeline configuration for all non-cached layers, add the following line to your contextConfigLocation (in web.xml):

```
<context-param>
   <param-name>contextConfigLocation</param-name>
   <param-value>
    <!-- generic Geomajas context (normal pipeline defintions) -->
    classpath:org/geomajas/spring/geomajasContext.xml

    <!-- delay filtering -->
    classpath:org/geomajas/plugin/vendorspecificpipeline/DefaultDelaySecurityPi

    <!-- layers and maps included through geomajasExampleContext.xml -->
    </param-value>
</context-param>
```

To override the default pipeline configuration for all cached layers, add the following line to your contextConfigLocation (in web.xml):

```
<context-param>
   <param-name>contextConfigLocation</param-name>
   <param-value>
    <!-- generic Geomajas context (normal pipeline defintions) -->
    classpath:org/geomajas/spring/geomajasContext.xml

    <!-- delay filtering -->
    classpath:org/geomajas/plugin/vendorspecificpipeline/DefaultCachedAndDelaySe

    <!-- layers and maps included through geomajasExampleContext.xml -->
    </param-value>
</context-param>
```

To override a single layer pipeline definition for a non-cached layer, add the following line to your contextConfigLocation (in web.xml):

```
<context-param>
```

```
   <param-name>contextConfigLocation</param-name>
   <param-value>
    <!-- generic Geomajas context (normal pipeline defintions) -->
    classpath:org/geomajas/spring/geomajasContext.xml

    <!-- delay filtering -->
    classpath:org/geomajas/plugin/vendorspecificpipeline/DelaySecurityPipeline.x

    <!-- layers and maps included through geomajasExampleContext.xml -->
    </param-value>
</context-param>
```

and add the following bean to your Spring context:

To override a single layer pipeline definition for a cached layer, add the following line to your contextConfigLocation (in web.xml):

```
<context-param>
   <param-name>contextConfigLocation</param-name>
   <param-value>
    <!-- generic Geomajas context (normal pipeline defintions) -->
    classpath:org/geomajas/spring/geomajasContext.xml

    <!-- delay filtering -->
    classpath:org/geomajas/plugin/vendorspecificpipeline/CachedAndDelaySecurity

    <!-- layers and maps included through geomajasExampleContext.xml -->
    </param-value>
</context-param>
```

and add the following bean to your Spring context:

```
 <bean class="org.geomajas.service.pipeline.PipelineInfo">
  <property name="pipelineName"><util:constant static-field="org.geomajas.servi
  <property name="layerId" value="layerXxx" />
  <property name="delegatePipeline" ref="PIPELINE_CACHED_DELAY_SECURITY_GET_FEA
 </bean>
```