

LOD4WFS Documentation

Linked Open Data for Web Feature Services

User and Developer Manual

Application Version: BETA 0.4.3

Editor: Jim Jones

Date: 07.08.2014

Copyright © 2014 Institut für Geoinformatik, Universität Münster

User Manual

1. General Information

The LOD4WFS Adapter (Linked Open Data for Web Feature Services) is a service to provide access to Linked Geographic Data from Geographical Information Systems (GIS). It implements a service which listens to WFS requests and converts these requests into the SPARQL Query Language for RDF. After the SPARQL Query is processed, the LOD4WFS Adapter receives the RDF result set from the Triple Store, encodes it as a WFS XML document, and returns it to the client, e.g. a GIS (see **Figure 1**). This approach enables current GIS to transparently have access to geographic LOD data sets, using their implementation of WFS, without any adaptation whatsoever being necessary. In order to reach a higher number of GIS, the currently most common implementation of WFS has been adopted for the LOD4WFS Adapter, namely the OGC Web Feature Service Implementation Specification 1.0.0¹. It provides the basic functions offered by the WFS reference implementation, GeoServer.

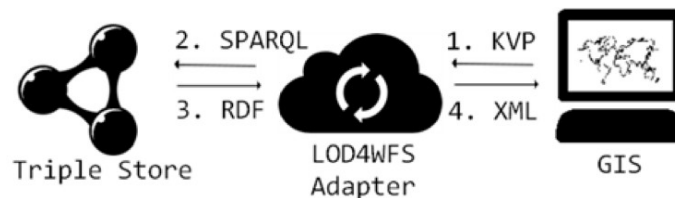


Figure 1. LOD4WFS Overview

The LOD4WFS Adapter enables access to geographic LOD data sets in two different ways: *Standard Data Access* and *Federated Data Accesses*.

1.1 Standard Data Access (SDA)

The Standard Data Access feature was designed in order to enable access to all geographic LOD data sets contained in a triple store. This feature basically works as an explorer, looking for geographic LOD data sets from a certain Triple Store and making them available via WFS. Due to the possibility of describing different types of geometries (polygons, lines, points) and many different coordinate reference systems, which are characteristic requirements of a WFS, we chose by default the GeoSPARQL Vocabulary as an input requirement for the Standard Data Access feature. Listing 1 shows how geometries and their related attributes are expected to be structured. The geometries are encoded as WKT literals and the attributes of Features are linked to the instance of the *geo:Geometry* class via RDF Schema and Dublin Core Metadata Element Set vocabularies. However, there are no constraints on which vocabularies or properties may be used for describing attributes.

¹ <http://www.opengeospatial.org/standards/wfs>

Listing 1. LOD-Data set Example: Turtle RDF encoding of a data set, together with its ID and Description.

```
@prefix geo: <http://www.opengis.net/ont/geosparql/1.0#>.
@prefix my: <http://ifgi.lod4wfs.de/resource/>.
@prefix sf: <http://www.opengis.net/ont/sf#>.
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

my:FEATURE_RECIFE a geo:Feature ;
    rdf:ID "2611606"^^xsd:integer ;
    dc:description "Recife"^^xsd:string ;
    geo:hasGeometry my:GEOMETRY_RECIFE .

my:GEOMETRY_RECIFE a geo:Geometry ;
    geo:asWKT "<http://www.opengis.net/def/crs/EPSG/0/4326>
POLYGON ((-35.0148559599999984 -8.0564907399999992, -34.9939074400000010
-8.0493884799999993, ... -35.0148559599999984 -8.0564907399999992))
"^^sf:wktLiteral .
```

1.1.1 Required Metadata

In order to make the data sets discoverable via the Standard Data Access feature, additional metadata must be added to the data sets. We make use of Named Graphs for this purpose. Every Named Graph in the LOD data source must contain only objects of the same feature type. This approach facilitates the discovery of Features², speeding up queries that list the Features available in the triple store. In case a Named Graph contains multiple Feature types, the features can be split into different layers using the Federated Data Access (see **Section 1.2**). Finally, each Named Graph needs to be described by certain RDF properties, namely *abstract*, *title* and *subject* from the Dublin Core Terms Vocabulary³. These properties help the adapter to classify all Features available in a Triple Store, so that they can be further on discovered by the WFS client through the WFS Capabilities Document (see **Listing 2**). Alternatively, the LOD4WFS Adapter could also use a query based on other RDF types to construct the Capabilities Document.

Listing 2. Named Graph Example.

```
@prefix dct: <http://purl.org/dc/terms/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

<http://ifgi.lod4wfs.de/graph/municipalities> dct:abstract "Municipalities of
the Brazilian Federal States."^^xsd:string .

<http://ifgi.lod4wfs.de/graph/municipalities> dct:title "Brazilian
Municipalities"^^xsd:string .
<http://ifgi.lod4wfs.de/graph/municipalities> dct:subject "municipalities
boundaries"^^xsd:string .
```

The LOD4WFS by default expects the Features' metadata to be encoded using these RDF properties, but they can be changed in the settings file (see **Section 2**).

² <http://www.opengeospatial.org/standards/sfa>

³ <http://dublincore.org/2012/06/14/dcterms.rdf>

1.2 Federated Data Access (FDA)

The Federated Data Access feature offers the possibility of accessing geographic LOD data sets based on predefined SPARQL Queries. Differently than the Standard Data Access, the Federated Data Access feature is able to execute SPARQL Queries to multiple SPARQL Endpoints, thus enabling WFS Features to be composed of data from different sources. As a proof of concept of what can be achieved, Listing 3 shows an example of a federated query, combining data from DBpedia and Ordnance Survey of Great Britain. The SPARQL Query is executed against the Ordnance Survey's SPARQL Endpoint, retrieving the GSS Code⁴ and geographic coordinates from districts of Great Britain – the coordinates are provided by the Ordnance Survey using the WGS84 lat/long Vocabulary, but this example converts them to WKT literals using the function CONCAT. Afterwards, the retrieved entries are filtered by matching the districts' names with DBpedia entries from the east of England, which are written in English language. The result of this SPARQL Query can be further on listed as a single WFS Feature via the LOD4WFS Adapter, thereby providing a level of interoperability between data sets that is currently unachievable by any implementation of WFS, whether using Shapefiles or geographic databases.

Listing 3. Federated Data Access - SPARQL Query Example.

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/resource/>
PREFIX wgs84: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX os: <http://data.ordnancesurvey.co.uk/ontology/admingeo/>

SELECT ?abstract ?name ?gss(CONCAT("POINT(", xsd:string(?long), " ",
xsd:string(?lat), ")") AS ?wkt)
WHERE
    {
        ?subject rdfs:label ?name .
        ?subject wgs84:lat ?lat .
        ?subject wgs84:long ?long .
        ?subject os:gssCode ?gss .
        ?subject a os:District
        SERVICE <http://dbpedia.org/sparql/> {
            ?entry rdfs:label ?place .
            ?entry dbpo:abstract ?abstract .
            ?entry dbpo:isPartOf dbp:East_of_England
            FILTER langMatches(lang(?place), "EN")
            FILTER langMatches(lang(?abstract), "EN")
            FILTER ( STR(?place) = ?name )
        }
    }
```

These predefined queries can be written and tested using the web interface. (see **Section 3**).

2. Setting up and starting the LOD4WFS Adapter

After downloading the LOD4WFS installation and unzipping it, you must input in the *settings.sys* file, located at the *settings* folder, the following information:

4 <http://data.ordnancesurvey.co.uk/ontology/admingeo/gssCode>

RDF properties to provide the necessary metadata for SDA based Features [GetCapabilities].
(see **Section 1.1**)

title

RDF property for the Features' title.

Default: <<http://purl.org/dc/terms/title>>

abstract

RDF property for the Features' abstract.

Default: <<http://purl.org/dc/terms/abstract>>

keywords

RDF property for the Features' subject.

Default: <<http://purl.org/dc/terms/subject>>

RDF properties in which the geospatial data is encoded [Geometry].

geometryPredicate

RDF property that connects the geometry instance to the encoded coordinates.

Default: <<http://www.opengis.net/ont/geosparql/1.0#asWKT>>

geometryClass

RDF class for the geometry objects.

Default: <<http://www.opengis.net/ont/geosparql/1.0#Geometry>>

predicatesContainer

RDF class for the Feature objects.

Default: <<http://www.opengis.net/ont/geosparql/1.0#Feature>>

featureConnector

RDF property to connect the Features to the geometries.

Default: <<http://www.opengis.net/ont/geosparql/1.0#hasGeometry>>

Application's settings [Server].

defaultPort

Port in which the LOD4WFS will listen the requests

Default: 8088

SPARQLEndpointURL

SPARQL Endpoint address which the SDA Features are located.

Default: -

SPARQLDirectory

Directory where the FDA Features' queries will be stored.

Default: `sparql/`

Web Interface Settings [WebInterface].

PreviewLimit

Maximum number of FDA Features' records via the web interface.
Default: 10

After the settings file is properly configured, you're ready to start the LOD4WFS adapter. The application is compressed in a single jar file (lod4wfs_release-x.x.jar) located in the root directory and can be started in the following command:

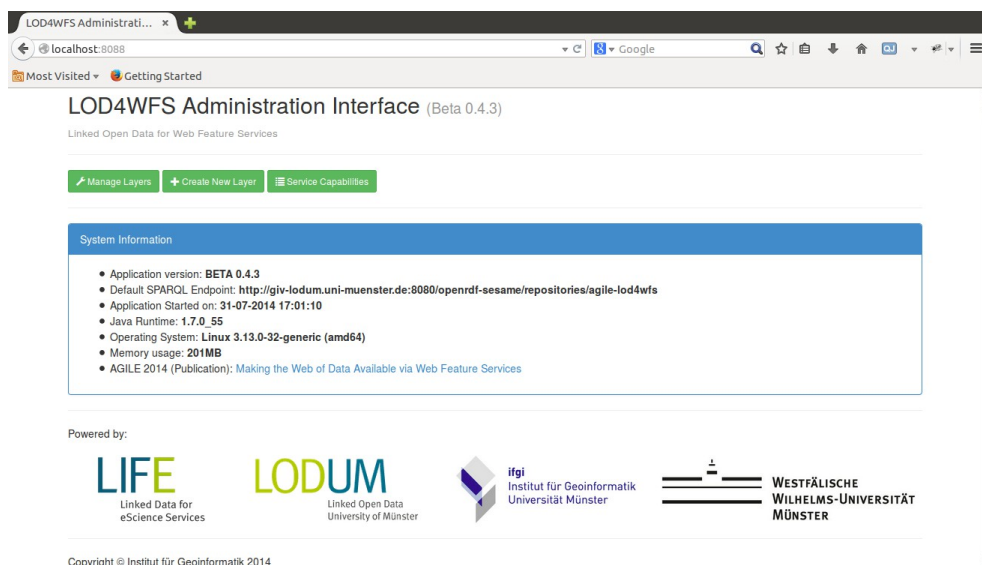
```
java -Xmx1g -jar lod4wfs_release-x.x.jar
```

Note: The parameter `-Xmx` reserves the amount of heap space necessary for the application. So, if you're dealing with large geometries, it might be necessary to increase the heap space, e.g. `-Xmx5g`. Since it is totally dependent on the geometries size, there is no optimum heap space size. For better maintainability, it is recommended to create a service in your operating system to manage the process created with command above mentioned.

3. The Web Administration Interface

The Web Administration Interface provides a web-based platform for users manage system's settings and FDA based Features. Once the system is up and running, type in the browser the server's address and the port configured in the settings file (see **Section 2**), e.g.

<http://localhost:8088/>



3.1 Creating a New Feature

In the Web Interface, click on the button "Create New Layer". A form with the following fields will be displayed.

SPARQL Endpoint

Endpoint address where the new Feature should be retrieved.

Feature Name

Name of new Feature, without spaces or special characters.

Title

Feature's title. Will be displayed in the capabilities document.

Abstract

A short explanation of the Feature. Will be displayed in the capabilities document.

Key-words

Key-words that describe the Feature. Will be displayed in the capabilities document.

Geometry Variable

Variable in the SPARQL Query that contains the geometry coordinates (Either GML or WKT).

SPARQL Query

Query returning the Feature's geometries and related attributes.

LOD4WFS Administration Interface (Beta 0.4.3)

[Home](#)
[Manage Layers](#)

New SPARQL-Based WFS Layer

SPARQL Endpoint	<input type="text" value="http://dbpedia.org/sparql"/>
Feature Name	<input type="text" value="feature_name_without_spaces"/>
Title	<input type="text" value="Feature Title"/>
Abstract	<input type="text" value="Short abstract of the feature"/>
Key-words	<input type="text" value="feature, key-words"/>
Geometry Variable	<input type="text" value="?wkt"/>
SPARQL Query	<pre> PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> PREFIX dbo: <http://dbpedia.org/ontology/> SELECT ?s ?lat ?long (concat('POINT(', ?long, ' ', ?lat, ')') AS ?wkt) WHERE { ?s <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> dbo:Place . ?s geo:lat ?lat . ?s geo:long ?long }</pre>

Validate

After filling in all fields, press the button **Validate**. If all fields were properly filled in and the SPARQL Query is valid a preview will be shown. To store the Feature, press **Save**.

3.2 Updating and Deleting Existing Features

In the web interface, click on **Manage Layers**, and a list with the available Features will be displayed. Every record in the list has the following buttons, which can be used to edit and delete Features:



Edit

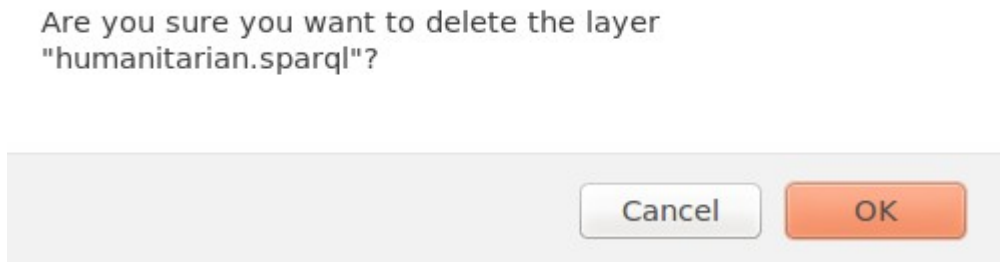
Delete

Layers available (7)					
Title	Abstract	Keywords	Endpoint		
HXL Repo	no abstract	no keywords	http://hxl.humanitarianresponse.info/sparql		
10k entries in DBpedia	10k random places in DBpedia. Retrieved via SPARQL.	dbpedia, places	http://dbpedia.org/sparql		
Piracy Data	no abstract	no keywords	http://semanticweb.cs.vu.nl/lop/sparql/		
SPARQL_Cool-Countries	Countries where I would live in. Data from DBpedia and GIV-LODUM.	Cool, Countries	http://recife:8080/openrdf-sesame/repositories/lod4wfs		
Bogota Administrative Boundaries	Bogota Administrative Boundaries	bogota, boundaries	http://giv-lodum.uni-muenster.de:8080/openrdf-sesame/repositories/agile-lod4wfs		
GSS Codes and Unit ID of GB	no abstract	no keywords	http://data.ordnancesurvey.co.uk/datasets/os-linked-data/apis/sparql		
no title given	no abstract given	no keywords given	http://dbpedia.org/sparql		

Clicking on Edit open a similar form to the New Feature form. There you can update all fields, with the exception of the Feature's name, regarding the selected Feature and validate its consistency. After you finished the changes click on **Preview**, if the Feature is valid (required fields and a resolvable SPARQL Query) result set is satisfactory click on **Save** to save the changes.

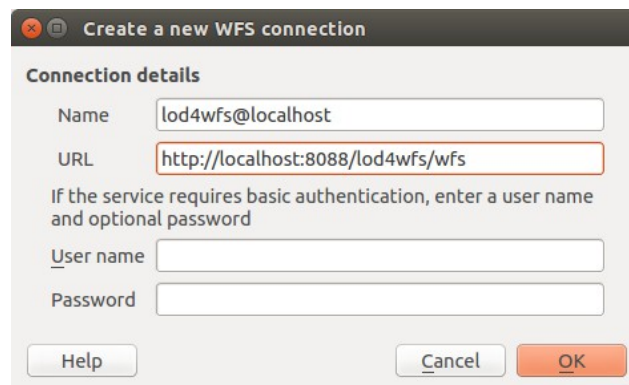
To delete a Feature simply click on the Feature and a dialog will ask you to confirm the Feature to

be delete. Afterward, press 'Yes'.

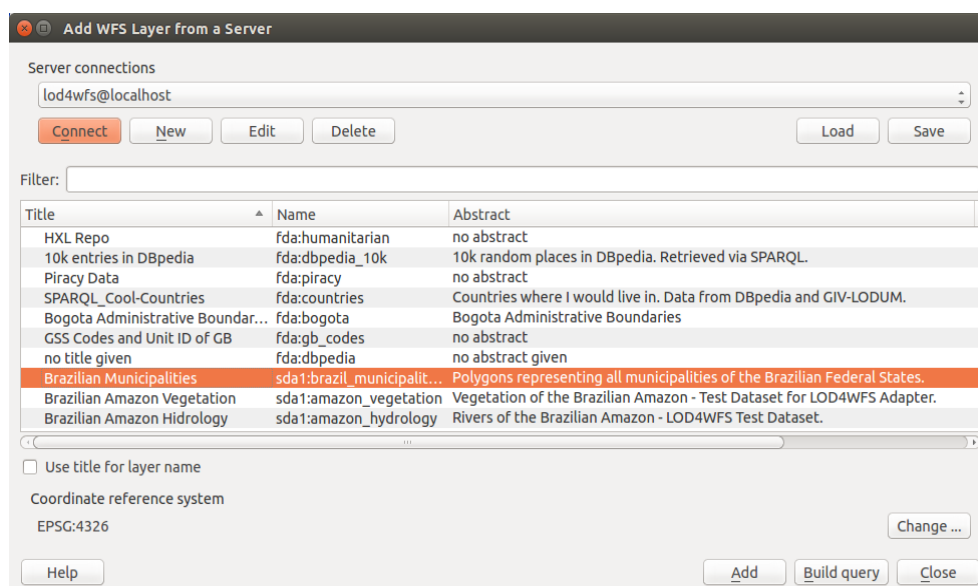


4. Using the LOD4WFS Adapter with QGIS 2.4.0

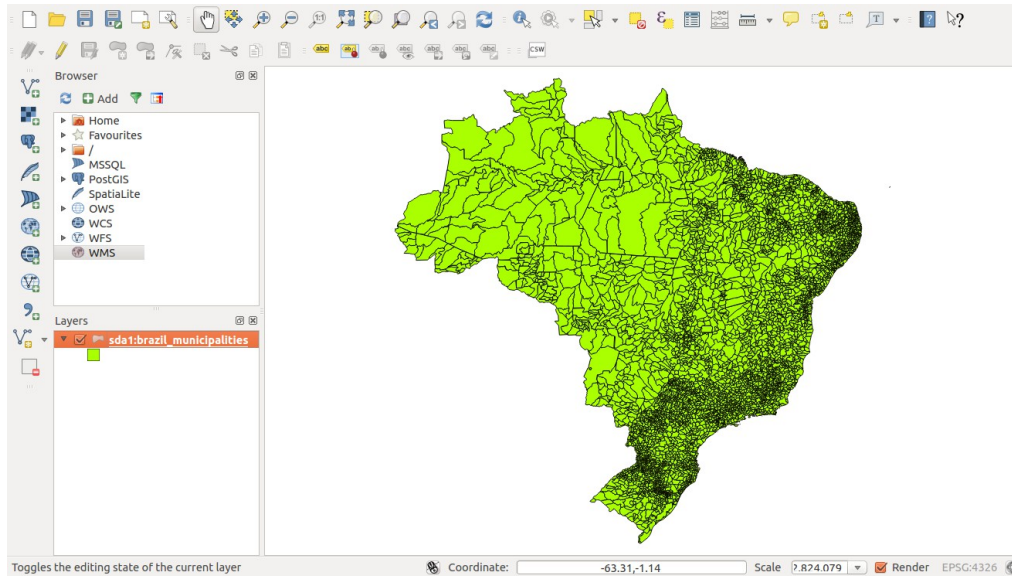
In QGIS go to **Layer > Add WFS Layer...** and click on **New**. Enter a [any] name for the service and the service address, and then press **OK**. No user or password is required.



Back to the **Add WFS Layer** screen, select the service you just created and press **Connect**. Then you receive a list SDA and FDA Features (separated by their name spaces) available in your service.



Click on the button **Add** and the Feature will be loaded into your QGIS environment together with its table of contents.

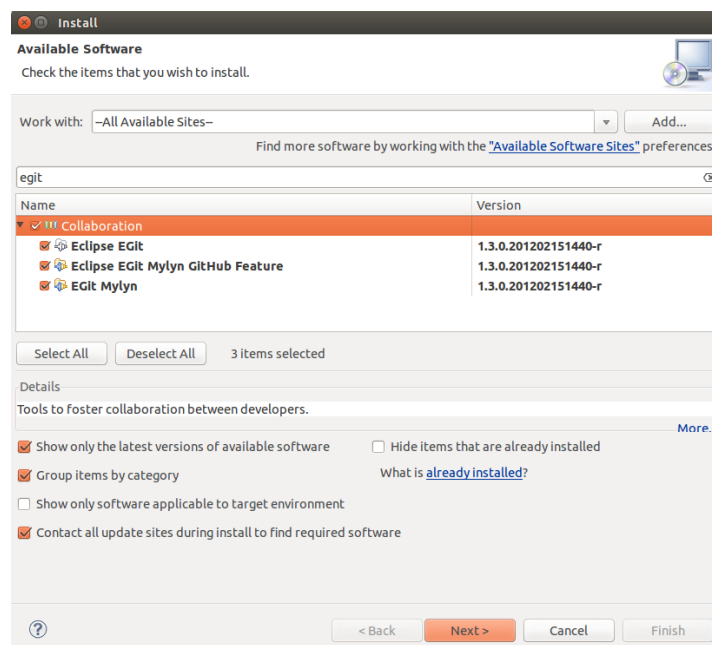


5. Developer Manual

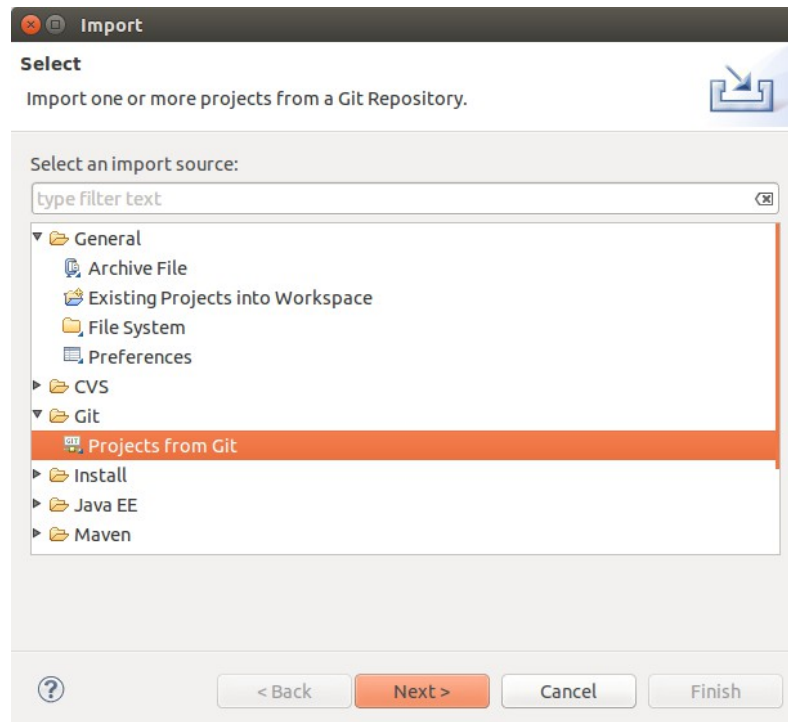
5.1 Source Code Installation

For installing the source code, it is assumed in this section that the Eclipse IDE is previously installed.

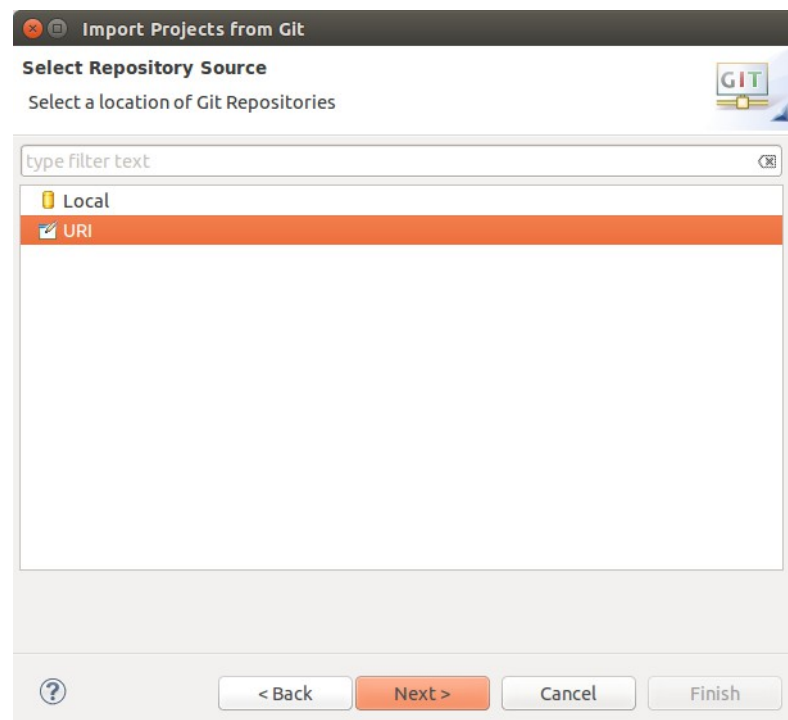
In Eclipse, go to **Help > Install New Software...** and search for the plug-in **egit**. Select the **Collaboration** check box, press **Next** and accept the license. You may be asked to restart your Eclipse.



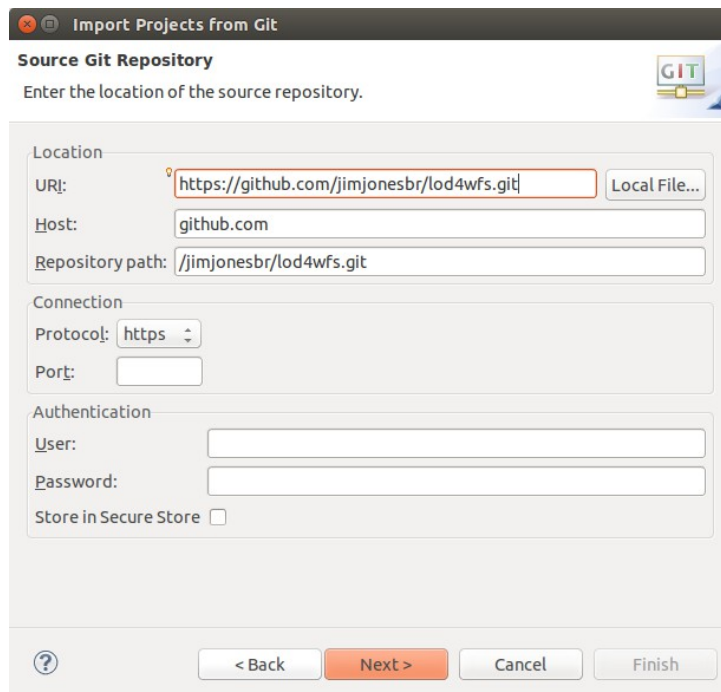
Now go to **File > Import...** and select **Git > Projects from Git** and press **Next**



Select **URI** and press **Next**.

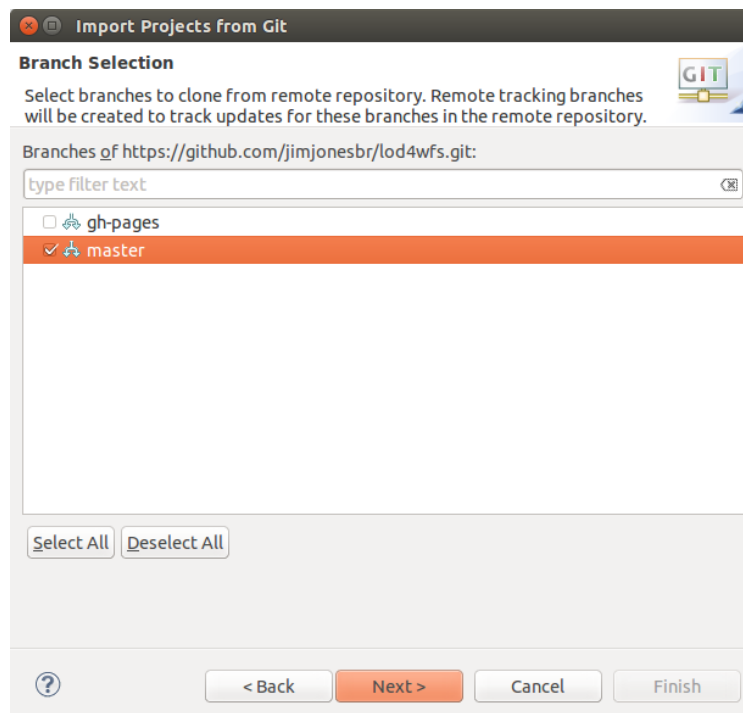


In the URI text field, place the following address:
`https://github.com/jimjonesbr/lod4wfs`



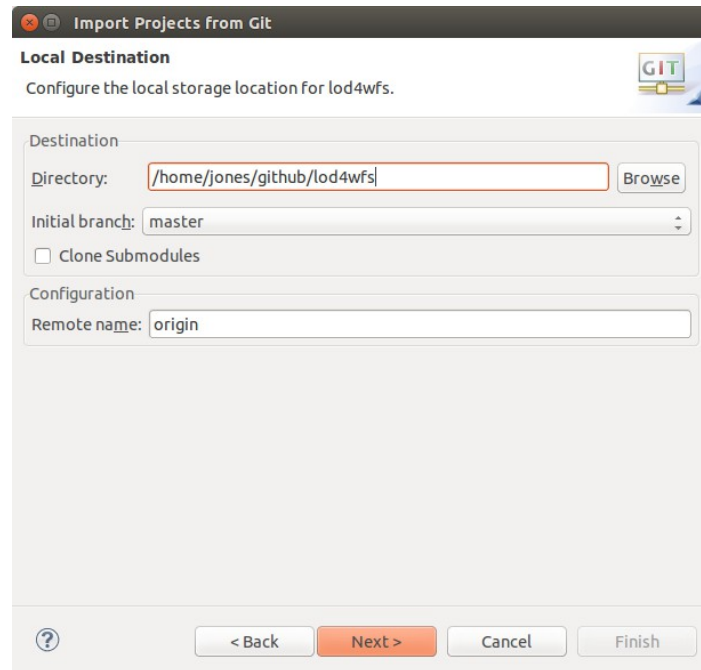
The screenshot shows the 'Import Projects from Git' dialog box, specifically the 'Source Git Repository' step. The dialog has a title bar with a close button and the text 'Import Projects from Git'. Below the title bar, there's a section titled 'Source Git Repository' with a subtitle 'Enter the location of the source repository.' and a small Git logo. The main area is divided into three sections: 'Location', 'Connection', and 'Authentication'. In the 'Location' section, the 'URI' field is highlighted with a red border and contains the text 'https://github.com/jimjonesbr/lod4wfs.git'. To its right is a 'Local File...' button. Below the URI field are fields for 'Host' (containing 'github.com') and 'Repository path' (containing '/jimjonesbr/lod4wfs.git'). The 'Connection' section has a 'Protocol' dropdown set to 'https' and an empty 'Port' field. The 'Authentication' section has 'User' and 'Password' fields, and a 'Store in Secure Store' checkbox which is unchecked. At the bottom of the dialog are four buttons: a help button (question mark), '< Back', 'Next >' (highlighted in orange), 'Cancel', and 'Finish'.

Select the branch **master** and press **Next**



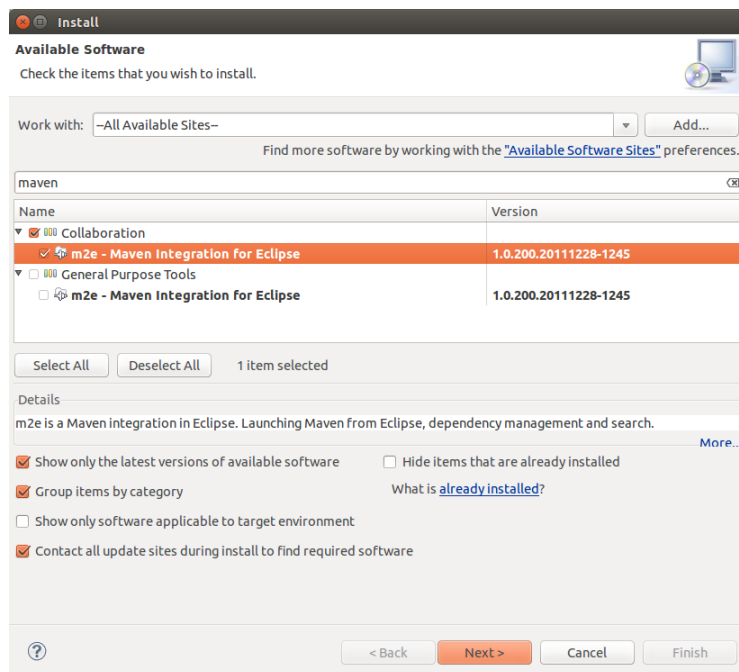
The screenshot shows the 'Import Projects from Git' dialog box, specifically the 'Branch Selection' step. The dialog has a title bar with a close button and the text 'Import Projects from Git'. Below the title bar, there's a section titled 'Branch Selection' with a subtitle 'Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.' and a small Git logo. The main area shows the text 'Branches of https://github.com/jimjonesbr/lod4wfs.git:' followed by a search bar labeled 'type filter text'. Below the search bar is a list of branches: 'gh-pages' and 'master'. The 'master' branch is selected, indicated by a checkmark and an orange highlight. At the bottom of the list are two buttons: 'Select All' and 'Deselect All'. At the bottom of the dialog are four buttons: a help button (question mark), '< Back', 'Next >' (highlighted in orange), 'Cancel', and 'Finish'.

Choose a place to store your local repository and press **Next**.



Select **Import existing projects** and press **Next** and then **Finish**.

Once the source code is downloaded, the next step is to install the application dependencies. The easiest way to do so is via the maven plug-in for Eclipse. To install maven go to Help > Install new Software... and search for Maven. As shown in the image below, select the Collaboration check box and press **Next**, accept the License and press **Finish**.



Now go to the pom.xml file on the project's root, right mouse click and go to Run As > Maven Install and your system is ready to use.

To start the application, simply run the class **de.ifgi.lod4wfs.web.Start** as a Java Application. After you start the adapter, the following message should appear in your console window:

```
LOD4WFS Adapter (Linked Open Data for Web Feature Service) BETA 0.4
Institut für Geoinformatik, Westfälische Wilhelms-Universität Münster
http://ifgi.uni-muenster.de/
```

```
Startup time: 29-07-2014 17:53:54
Port: 8088
```

6. Supported Operations

The LOD4WFS Adapter supports the following operations:

Operation	Description
GetCapabilities	Generates a metadata document describing a WFS service provided by server as well as valid WFS operations and parameters
DescribeFeatureType	Returns a description of Feature types supported by a WFS service
GetFeature	Returns a selection of Features from a data source including geometry and attribute values

6.1 GetCapabilities

The GetCapabilities operation is a request to a WFS server for a list of the operations and services, or capabilities, supported by that server.

Parameter	Required	Description
service	Yes	Always "WFS"
version	Yes	Currently only supporting 1.0.0
request	Yes	Always "GetCapabilities"

GetCapabilities request example:

<http://localhost:8088/lo4wfs/wfs/?service=wfs&version=1.0.0&request=GetCapabilities>

6.2 DescribeFeatureType

DescribeFeatureType requests information about an individual feature type before requesting the

actual data. Specifically, the operation will request a list of features and attributes for the given feature type, or list the feature types available.

The expected parameters for the DescribeFeatureType request are:

Parameter	Required	Description
service	Yes	Always “WFS”
version	Yes	Currently only supporting 1.0.0
request	Yes	Always “DescribeFeatureType”
typeName	Yes	Name of the Feature type to describe

DescribeFeatureType request example:

<http://localhost:8088/ld4wfs/wfs?service=wfs&version=1.0.0&request=DescribeFeatureType&typeName=namespace:feature>

6.3 GetFeature

The GetFeature operation returns a selection of Features from the data source.

The expected parameters for the GetFeature request are:

Parameter	Required	Description
service	Yes	Always “WFS”
version	Yes	Currently only supporting 1.0.0
request	Yes	Always “GetFeature”
typeName	Yes	Name of the selected Feature.
srsName	No	Spatial Reference System of the selected Feature, e.g. “EPSG:4326”
outputformat	No	<i>See supported output formats (if ignored, GML2 is assumed).</i>
format_options	No	<i>See supported format options.</i>

6.3.1 Output Formats

The GetFeature request currently supports the following output formats:

Format	Syntax	Description
GML2	outputformat=gml2	GML2. Default format described at the OGC WFS 1.0.0 specification.
GeoJSON	outputformat=text/javascript	GeoJSON

6.3.2 Output Format Options

The GetFeature request currently supports the following options for output formats:

Format	Syntax	Description
JSONP	format_options=callback:loadgeojson	Returns a JSONP object.
ZIP file	format_options=zip	Returns the Feature compressed in a ZIP file.

GetFeature request example:

http://localhost:8088/od4wfs/wfs?service=wfs&version=1.0.0&request=GetFeature&typename=namespace:feature&outputformat=text/javascript&format_options=callback:loadgeojson

7. Exceptions

Requests errors are reported in XML format, mostly according to the OGC WFS Specification⁵. The predefined exceptions are:

Exception	Description
ServiceNotProvided	Service not provided in the request.
ServiceNotSupported	Invalid service provided.
VersionNotProvided	Invalid WFS version provided.
OperationNotSupported	Invalid operation. <i>See Section 6.</i>
FeatureNotProvided	Invalid feature provided.
InvalidOutputFormat	Invalid output format provided. <i>See Section 6.3.1</i>
InvalidOutputFormatOption	Invalid output format provided. <i>See Section 6.3.2</i>

⁵ <http://www.opengeospatial.org/standards/wfs>