Last edited by **Ondřej Vašíček** 2 weeks ago

1. Configuration

Configuration changes only take effect after restarting the adapter. Output filter class changes currently only take effect after rebuilding the adapter.

1) Triplestore

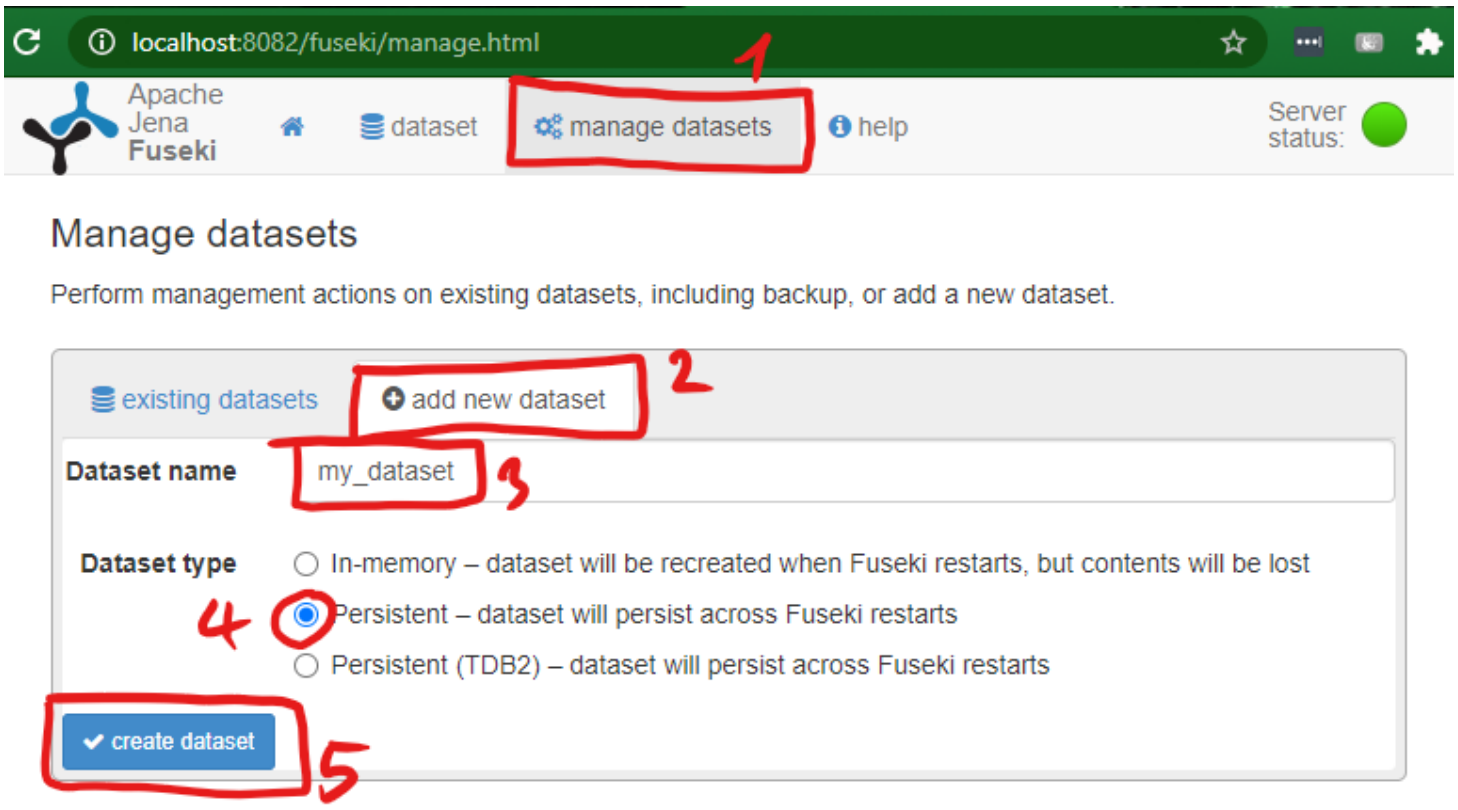
Configure host and port in conf/TriplestoreConf.ini

conf_example > TriplestoreConf.ini

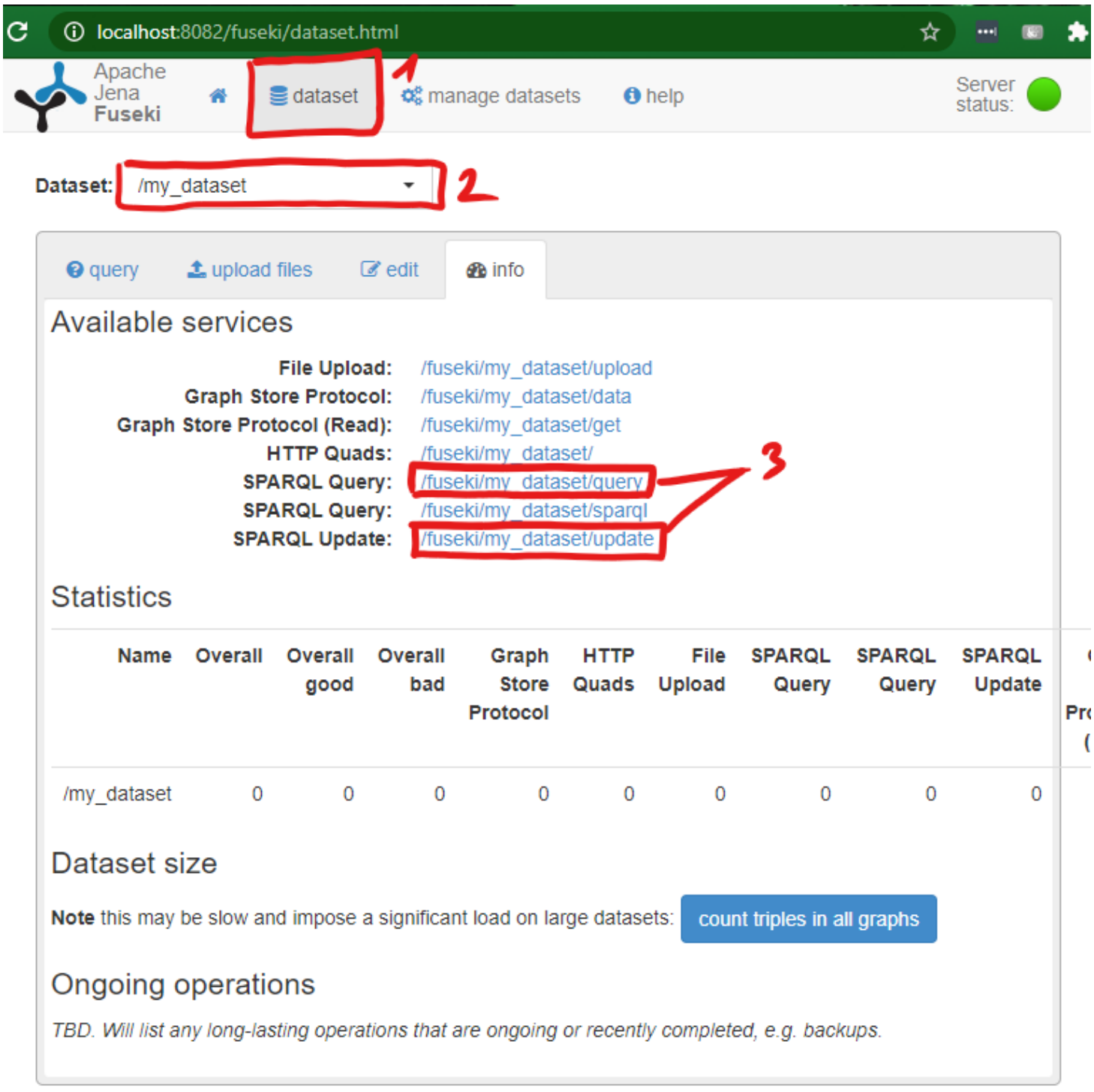
```
## THIS IS A EXAMPLE CONFIGURATION FILE
##

--module=http
    # port to run the triplestore on
    jetty.http.port=8082
    # host to run the triplestore on
    jetty.http.host=localhost

--module=home-base-warning
--module=ext
--module=server
--module=jsp
--module=resources
--module=deploy
--module=jstl
--module=websocket
```

1.1) Creating a persistent dataset

First create dataset using the Fuseki Web UI. Set the dataset name at point 3.



Look up the endpoints of your new dataset. Use these endpoints to configure the Analysis and the Compilation adapters.



2) Compilation

In conf/VeriFitCompilation.properties

1. configure host and port
2. set the triplestore host and port according to your triplestore configuration
3. set the triplestore endpoint URIs based on your triplestore info (from 1.1)
4. name your sparql graph
5. set to true if you are using a persistent dataset
6. authentication settings
7. automatic deletion of old resources

conf_example > VeriFitCompilation.properties

```
1  ## THIS IS A DEFAULT/EXAMPLE CONFIGURATION FILE
2  ##
3  ################################################################################
4  ## URL CONFIGURATION
5  ################################################################################
6  adapter_host=http://localhost
7  adapter_port=8081
8
9  ################################################################################
10 ## TRIPLESTORE CONFIGURATION
11 ################################################################################
12 sparql_query=http://localhost:8082/fuseki/verifit_compilation/query
13 sparql_update=http://localhost:8082/fuseki/verifit_compilation/update
14 sparql_graph=http://verifit_compilation
15
16 ################################################################################
17 ## PERSISTENCY CONFIGURATION
18 ################################################################################
19 persist_sut_dirs=false
20
21 ################################################################################
22 ## AUTHENTICATION CONFIGURATION
23 ################################################################################
24 enable_authentication=false
25 username=user
26 password=user
27
28 ################################################################################
29 ## AUTOMATIC CLEAN UP CONFIGURATION
30 ################################################################################
31 keep_last_n_enabled=false
32 keep_last_n=100
```

3) Analysis

In conf/VeriFitAnalysis.properties

1. configure host and port
2. set the triplestore host and port according to your triplestore configuration
3. set the triplestore endpoint URIs based on your triplestore info (from 1.1)
4. name your sparql graph
5. authentication settings
6. automatic deletion of old resources

conf_example > VeriFitAnalysis.properties

```
## THIS IS A EXAMPLE CONFIGURATION FILE
##

################################################################################
## URL CONFIGURATION
################################################################################
adapter_host=http://localhost
adapter_port=8080

################################################################################
## TRIPLESTORE CONFIGURATION
################################################################################
sparql_query=http://localhost:8082/fuseki/verifit_analysis/query
sparql_update=http://localhost:8082/fuseki/verifit_analysis/update
sparql_graph=http://verifit_analysis

################################################################################
## AUTHENTICATION CONFIGURATION
################################################################################
enable_authentication=false
username=user
password=user

################################################################################
## AUTOMATIC CLEAN UP CONFIGURATION
################################################################################
keep_last_n_enabled=false
keep_last_n=100
```

3.1) Analysis tool definition

In conf/analysis_advanced/AnalysisTools/YourTool.properties

1. full path to the executable of your tool
2. default arguments to be always used with your tool (optional)
3. set to true to enable queuing for your tool (fifo one at a time)

conf_example > analysis_advanced > AnalysisTools > ExampleTool.properties

```
## Path to the tool executable
## IMPORTANT: Use double backslash on windows! (\\ instead of just \)
toolLaunchCommand=/full/path/to/executable.sh

## Arguments to always use on the command line when launching the tool
(e.g. to make the tools output readable by the
## adapter). These will always be placed as the first command line
parameter.
#toolSpecificArgs=--example

## If set to true, then only one AutomationRequest executing this
AutomationPlan will be running at a time and the
## remaining ones will be placed in a queue.
oneInstanceOnly=False
```

In conf/analysis_advanced/AnalysisTools/YourTool.rdf

1. uniquely name your automation plan (will be the last part of its URI)
2. define parameters for your automation plan
3. parameters need a unique name (unique within the automation plan), watch out for collisions with common adapter parameters
4. parameters whose values should be inserted to the command line need a commandlinePosition
5. occurrence determines whether a parameter is optional
6. value type gives information to clients on what value the parameter expects
7. launchSUT is a special input parameter along with SUTbuilCommand
8. default values can be specified for any parameter

conf_example > analysis_advanced > AnalysisTools > ExampleTool.rdf

```
<oslc_auto:AutomationPlan>
    <dcterms:identifier>example</dcterms:identifier> <!-- What the last part of this AutoPlan's URI will c
    <oslc_auto:usesExecutionEnvironment rdf:resource="https://url.to.your.tool.com"/> <!-- Non-functional.
    <dcterms:title rdf:parseType="Literal">Example Tool</dcterms:title>   <!-- Non-functional. Title of th
    <dcterms:description rdf:parseType="Literal">Used as an example.</dcterms:description> <!-- Non-functi
    <dcterms:creator rdf:resource="https://url.to.the.creator.com"/> <!-- Non-functional. Creator of the A
    <oslc_auto:parameterDefinition>
        <oslc_auto:ParameterDefinition>
            <oslc:name>arguments</oslc:name>
            <fit:commandlinePosition rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</fit:commandlin
            <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
            <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
            <dcterms:description rdf:parseType="Literal">Specify which arguments should be passed to the c
        </oslc_auto:ParameterDefinition>
    </oslc_auto:parameterDefinition>
    <oslc_auto:parameterDefinition>
        <oslc_auto:ParameterDefinition>
            <oslc:name>launchSUT</oslc:name>
            <fit:commandlinePosition rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2</fit:commandlin
            <oslc:defaultValue>True</oslc:defaultValue>
            <oslc:occurs rdf:resource="http://open-services.net/ns/core#Zero-or-One"/>
            <dcterms:description rdf:parseType="Literal">This parameter definitions tells the Automation P
        </oslc_auto:ParameterDefinition>
    </oslc_auto:parameterDefinition>
    <oslc_auto:parameterDefinition>
        <oslc_auto:ParameterDefinition>
            <oslc:name>SUTbuildCommand</oslc:name>
            <fit:commandlinePosition rdf:datatype="http://www.w3.org/2001/XMLSchema#int">3</fit:commandlin
            <oslc:defaultValue>True</oslc:defaultValue>
            <oslc:occurs rdf:resource="http://open-services.net/ns/core#Zero-or-One"/>
            <dcterms:description rdf:parseType="Literal">This parameter definitions tells the Automation P
        </oslc_auto:ParameterDefinition>
    </oslc_auto:parameterDefinition>
</oslc_auto:AutomationPlan>
```

3.2) Output filter definition

In conf/analysis_advanced/PlugiFilters/YourFilter.properties

1. set the name of your filter class
2. specify the name of an Automation Plan which your filter is meant for

conf_example > analysis_advanced > PluginFilters > ExamplePluginFilter.properties

```
# do not change this value
implements=cz.vutbr.fit.group.verifit.oslc.analysis.outputFilters.IFilter

# class with implementation of the filter -- change the last part only to match your .java file
class=pluginFilters.customPluginFilters.ExamplePluginFilter

# says which tool is this filter meant for -- needs to match an AutomationPlan identifier
tool=example
```

In conf/analysis_advanced/PlugiFilters/YourFilter.class

1. uniquely name your class implementing the required interfaces
2. uniquely name your filter
3. other filters can be used within your filter
4. optionally do some processing with the inputs of your filter
5. optionally create new contributions with unique IDs

conf_example > analysis_advanced > PluginFilters > ExamplePluginFilter.java

```java
public class ExamplePluginFilter implements IFilter, IExtension {
    final String name = "example";

    public void filter(List<Map<String, String>> inoutContributions) {
        // run Contributions through one of the builtin parsers first to load stdout and
        new AddStdoutAndStderrValues().filter(inoutContributions);

        // look for data race detection reports in stdout
        Boolean dataRaceFound = false;
        for (Map<String, String> contrib : inoutContributions)
        {
            String title = contrib.get("title");
            if (title.equals("stdout"))
            {
                String contentsOfTheStdout = contrib.get("value");
                if (contentsOfTheStdout.contains("Data race detected at"))  // check if
                {
                    dataRaceFound = true;
                }
            }
        }

        // create a contribution representing the result (based on the stdout contents)
        Map<String, String> contrib = new HashMap<String, String>();
        contrib.put("id", "example_id");    // use any identifier you want (can be used
        contrib.put("title", "DataRaceDetected");       // use any name you want
        contrib.put("description", "Holds the result of data race analysis.");
        contrib.put("value", dataRaceFound.toString());
        contrib.put("valueType", "http://www.w3.org/2001/XMLSchema#boolean");
        inoutContributions.add(contrib);
    }

    @Override
    public String getName() {
        return name;
    }
}
```