

Getting started with the NonStopGov SOAPClient

The NonStopGov SOAPClient is a middleware utility implemented using Apache Tomcat, an open source servlet engine. It is typically used to periodically load data from disparate datasources and send that data to a central server. The NonStopGov SOAPClient can only run on a system with a Java Development Kit (JDK) installed. An important consideration when choosing where to install a NonStopGov SOAPClient is the access it will need. Typically, the NonStopGov SOAPClient must be able to reach the datasource it will be extracting data from and must also have internet access to send that data to a central server.

For a complete installation please follow the following steps.

1. Install Java

If the machine that will host the SOAPClient does not currently have a JDK (Java Development Kit (v1.4 or later) installed, one will have to be installed. An installation file for windows systems can be found on our FTP server at the following location –

ftp://nsgdownload.tagish.net/FSA/jdk-1_5_0_11-windows-i586-p.exe

Installation files for other operating systems are available on the Sun website at http://java.sun.com/javase/downloads/index_jdk5.jsp.

The windows installer has a simple wizard interface. On the first page accept the license agreement, on the second page leave the components to be installed to the default setting and choose a directory to install the JDK into. Make a note of the directory you choose as it will be needed later for configuration purposes. Follow the same procedure to install the second half of the Java system, the Java Runtime Environment (JRE) and finally allow the Java web browser plug-in to be installed (though this is not strictly necessary) .

2. Install Tomcat

A preconfigured instance of the Apache Tomcat servlet engine incorporating the NonStopGov SOAPClient is available on our FTP server at the following location –

<ftp://nsgdownload.tagish.net/FSA/jakarta-tomcat-5.0.28.zip>

This instance of Tomcat already has the SOAPClient deployed in it, and simply needs to be unzipped to a working directory. Once Tomcat is unpacked to a directory, two environment variables must be set. On Windows systems, environment variables can be found at Control Panel -> System -> Advanced -> Environment Variables. Add a variable called JAVA_HOME with its value set to the path under which you installed your JDK (and made a note of) earlier. Add a second variable called CATALINA_HOME with its value set to the path to which you have just unzipped your Tomcat installation.

3. Configure Tomcat to run as a service (Windows)

For Windows based operating systems, you can install Tomcat as a service. Installing Tomcat as a service is strongly recommended and means that it will start running automatically at machine start-up and will run as a background process. This is of particular importance when uploads of inspection data are scheduled to be sent at periodic intervals, requiring Tomcat to be up and running at all times. To install the Tomcat service, open a command line window, navigate to the /bin directory of your Tomcat installation and run the following command:

```
service.bat install NSGLASOAP
```

This will create a service with the name Apache Tomcat NSGLASOAP (or any other name you choose to give it) which will then appear on your services menu, Start Menu -> Administrative Tools -> Services, where it can be stopped, started or restarted when required.

4. Configuring data uploads in the SOAPClient

To start the SOAP Client configuration tools, navigate to the following web page using your browser:

http://<machine name>:8080/ns gla_soap/soapclient.client?service=foodsafety

Hitting the URL above will take you to the welcome page of the SOAPClient, which gives a brief description of what each section of the tool allows you to setup and shows the menu of links to these sections on the left hand side of the page. The first section to look at is the “Destination” page of the tool, where you can enter the server path to which your uploads will be sent and a password for additional security. The server path will be pre-configured to the London test server and does not need changing at this point. A password will be provided in due course.

The next step is to configure an upload category. An upload category is simply a query against a datasource on your own system which is run periodically to extract inspection records and send them to the central server. In the “Food safety inspections” page of the Load data section, a list of current upload categories will be displayed, which at this point should have only one entry. The listed category, id: ‘ID’ and name: ‘Food Safety Inspections’, is a demonstration category which can be edited to set up your uploads. Hit the edit link on the category’s row to launch the category wizard. Before going in to detail, please **note that each item to be configured in the wizard has additional help text associated with it, which can be found by hovering your mouse pointer over, or clicking on, the question mark icon next to each field label.** The help text gives more information about what a field relates to and how it should be used.

On the first page of the wizard there are three fields to fill in, the first ‘id’ field, currently set to ‘ID’, should be over written with your four digit local authority code. The display name is simply a label to help you identify your category (the SOAPClient allows any number of categories to be setup, though in this case only one is required). The display name is not uploaded to the server, only stored locally for your use so it can be set to any value you like or left as the default ‘Food Safety Inspections’. The datasource type drop down tells the SOAPClient which type of datasource it will be extracting your inspections data from. The client can interact with any database using ODBC (or JDBC) as well as with CSV files or XML feeds of any format, though configurations for common formats such as RSS and Atom feeds are built in to save time. Select the datasource you will be using and hit next to proceed.

Page two of the wizard is where the query that will be used to look up inspection data in your system is configured. All the fields expected according to the agreed schema for FSA inspection records are listed and next to each you should fill in where in your datasource data for that field can be found. If you selected a database then each field should be filled in with the name of a corresponding database column, and the table or view in which those columns can be found should be entered in the corresponding field lower down the page. If your datasource is CSV based, each field should correspond with a column header included on the first line of the CSV file. If your datasource is XML based, the XML elements relating to each field should be entered using XPath syntax¹. For both SQL and XML datasources, a ‘where’ field allows the query to narrow down the exact data set it extracts.

The final page of the wizard asks for full details of the datasource to be used. The exact details asked for will differ depending on the type of datasource selected on page one, but all these details should be readily available – file locations, feed URLs, database logins etc. Also on this page, the schedule of uploads is configured for your new category. If you choose to run a regular, automatic upload of data (uploads can also be sent manually at any time) you simply have to set a time for the first upload to be processed and then an interval length at which the upload will be reprocessed after that initial load. For example, setting the execution start time to

¹ The file XPath tips.txt bundled with the SOAPClient contains basic guides to the kinds of XPath strings that will be needed in an upload category.

11pm and the execution interval to 12 hours would cause uploads to be made automatically twice day, at 11pm and 11am. We recommend that uploads are scheduled at 24hr intervals each evening/night, before 7AM. Leaving the execution interval set to zero will mean the category is not scheduled for automatic uploads at all.

On completing the wizard you will be returned to the 'Food safety inspections' page showing your list of categories, which will now have your newly configured category listed. Next to its id and name you will see four actions that can be taken against the upload category.

To change any settings, hit 'edit' which will launch the category wizard again and show the current settings of your category.

To delete the category completely, hit the remove link. You will be asked to confirm your decision as removed categories can not be restored.

5. Testing uploads

To test the query in your category, hit the 'test' link. This causes the SOAPClient to execute the query against the datasource but rather than send the extracted data to the server, the extract is saved to a file on your system which you can view through the link provided to verify that the correct data is being selected in the correct fields.

Alternatively, to perform a manual upload of your category, hit send. This will process the query and sent it to the server immediately. It is worth attempting an upload of your category even at this early stage, if just to check that your SOAPClient has proper access to the central server. After hitting the send link and closing the upload window which is launched, check the SOAPClient log file, stdout.log in the logs subdirectory of your Tomcat installation. A successful upload will be show in the log with an entry similar to the following

```
- 08-Feb-2007 14:40:49 - Adhoc Run
- 08.02.2007 14:40 - Processing fs_inspections, id: 6598
- 5 items sent
```

If there is a problem connecting to the server, an error message will be logged e.g.

```
- 08-Feb-2007 14:38:59 - Adhoc Run
- 08.02.2007 14:38 - Processing fs_inspections, id: 6598
- java.net.UnknownHostException: eforms.yourondon.gov.uk
javax.xml.soap.SOAPException: java.net.UnknownHostException: eforms.yourondon.gov.uk
  at org.apache.axis.soap.SOAPConnectionImpl.call(SOAPConnectionImpl.java:95)
  at soapclient.send.writeSOAPBody(send.java:1319)
  at soapclient.send$scheduleTasks.processClientData(send.java:543)
...
```

The error message will change depending on the issue or networking error (typically firewall issues) which has caused the fault.