Oracle® Enterprise Data Quality for Product Data

Web Service Access to Oracle DataLens Servers Interface Guide

Release 11g R1 (11.1.1.6)

E29146-01

August 2012

This document describes the Oracle Enterprise Data Quality for Product Data Web Access to Oracle DataLens Servers and contains the following:

Overview

Web service access is provided to the Oracle DataLens Server as a Document-Literal Web Service. The following sections describe how to set up this access and the Web Service Operations that you can use to interface with Enterprise DQ for Product (EDQP) Data Service Applications (DSAs) including examples.

Obtaining the EDQP WSDL Document

To integrate with an Enterprise DQ for Product DSA as a Web Service, you need software that will interface with the Oracle DataLens Web Services. There are a number of tools available to generate this software from a Web Services Description Language (WSDL) document, which is an XML format for describing network services. You can view the WSDL for the Oracle DataLens Web Services by using an Internet browser.

Browse to the following Web site:

```
http://server:port/datalens/ws/Processor?wsdl
```

Where *server:port* is the hostname and port number of your Oracle DataLens Server.

This displays the WSDL document, which can be saved by right-clicking in the document in the browser, selecting **View Source,** and then saving the file from within your browser. For instance, the file can be saved as `Processor.wsdl`.

Client Web Service Software

For your Web Service clients, client-side software can be generated from this WSDL document to access the Oracle DataLens Server.

Communicating with a DSA

There is a single Service called "ProcessorService", which uses a PortType (Interface) called "Processor".

The following three Oracle DataLens Web Services Operations can be used to process data and communicate with a DSA by creating a job:

`ProcessorList`
This requires an input array of strings, or multiple lines of data, and returns an output array of strings. One DSA job is created.

`ProcessorOneLine`
This requires a single string of input, or one line of data, and returns a single string of output. If you have several lines of data, use `ProcessorList` to process them because a single DSA job is created rather than the multiple DSA jobs that would be created using the `ProcessorOneLine` operation for each line of data. For example, if you have four lines of data to process, the `ProcessorList` creates one DSA job while the `ProcessorOneLine` creates four separate DSA jobs.

`ProcessorDB`
This requires a database query (defined in the Transform Map) and returns a job Id of the DSA Job that handled the request. The output is assumed to be a database update, e-mail, or FTP. One DSA job is created.

Points to Consider

- Reviewing the EDQP WSDL will help you to understand what operations and parameters can be used to communicate with DSAs.
- Simple Object Access Protocol (SOAP) is used to communicate with DSAs. For more information about SOAP, see the W3C SOAP 1.1 Web site:

  http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
- You must provide the exact input fields that are defined in the DSA to ensure proper operation and avoid errors. For example, if the input nodes in the DSA are `Id`, `Category`, and `Description`, you must provide all three fields in your SOAP processing operation.
- You must separate input fields using a defined separator; the tab character "^" is recommended. For example, using the tab character to separate the `Id`, `Category`, and `Description` input fields:

  ```
  <lineOfData>3^2334^res, 37ohm, 39watt, 30%</lineOfData>
  ```
- The best practice is to define all of the operation parameters (even those that may be labeled as optional) to avoid errors. The EDQP WSDL and the DSA define the necessary parameters not the SOAP UI you are using; some SOAP UIs erroneously mark parameters as optional as in the examples in the following sections.

  The SOAP request structure is the same for each type of request though some of the parameters may not apply. For example, `processListRequest` and `processOneLineRequest` do not use the `dbparameters` parameter; `processDBRequest` does not use `line` or `linesofdata` parameters. The parameters used are dependent on the operation request.

  Additionally, defining all parameters allows the most information possible to be related to the DSA job, and then reviewed and verified in the Oracle DataLens Server Administration Web Page.

`processListRequest` and `processOneLineRequest` Operations

The difference between these two operations is that `processListRequest` takes an array of lines and `processLineRequest` takes a single line of data as a string. The transformed data is returned. This call is synchronous. The parameters are as follows:

`dsaName`
The name of the DSA to run on the Oracle DataLens Server.

`line` or `linesOfData`
One line or multiple lines of input or output data respectively.

`dbParameters`
Not used.

`priority`
The priority the DSA job will be given on the Oracle DataLens Server for processing; the default is Medium. The priority is set numerically as follows:

`3` for High
`2` for Medium (the default)
`1` for Low

`runtimeLocale`
A run time locale must be entered; the default locale of `en_US` should be entered if not known.

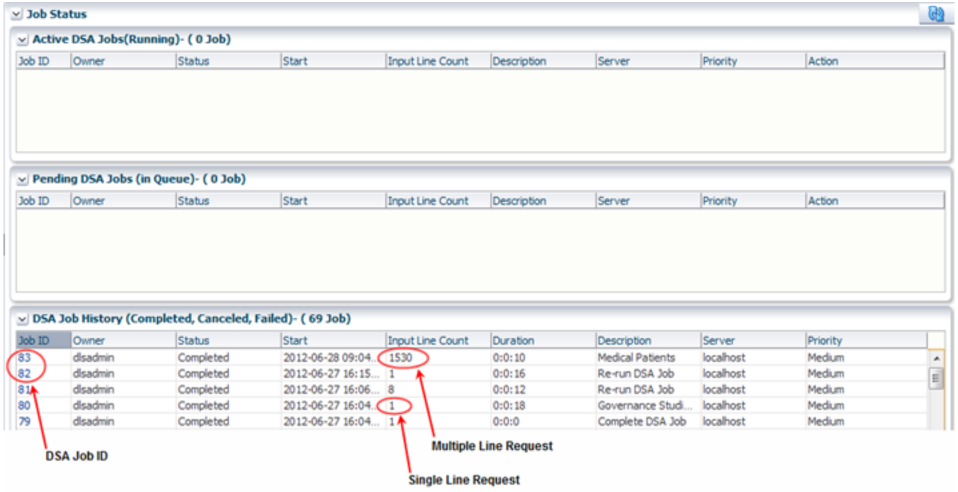
`fieldSeparatorChar`
A character that will be used to separate fields; the tab character "^" is recommended.

`application`
This application name initiated the client request to the server. The application name must be a valid LDAP entry in the WebLogic Server to avoid a failure. This name is used to accumulate server statistics on the Oracle DataLens Server Administration Web Page.

`description`
A description of a particular job and relates to a DSA Job ID.

The Oracle DataLens Server Administration Server Job Status Web page shows the DSA Job ID for both the `processListRequest` and `processLineRequest` operations as in Figure 1.

Figure 1 DSA Jobs with Single and Multiple Line Input



Job Status

Active DSA Jobs(Running)- (0 Job)

Job ID	Owner	Status	Start	Input Line Count	Description	Server	Priority	Action

Pending DSA Jobs (in Queue)- (0 Job)

Job ID	Owner	Status	Start	Input Line Count	Description	Server	Priority	Action

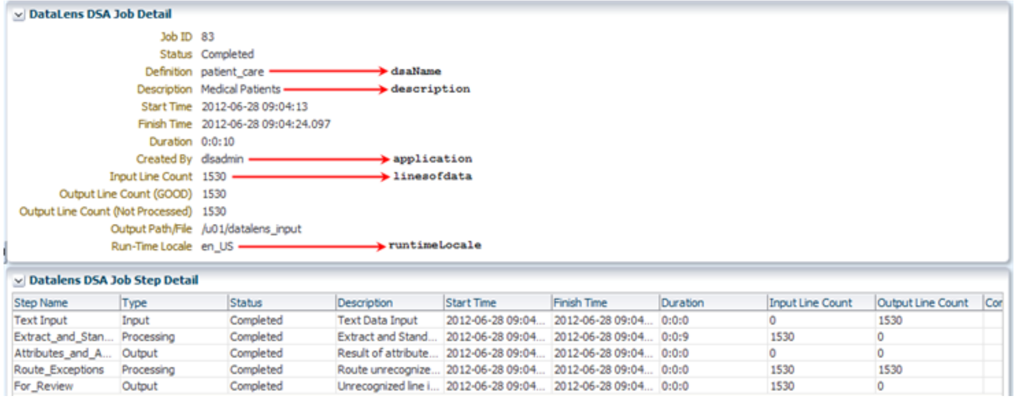
DSA Job History (Completed, Canceled, Failed)- (69 Job)

Job ID	Owner	Status	Start	Input Line Count	Duration	Description	Server	Priority
83	dlsadmin	Completed	2012-06-28 09:04...	1530	0:0:10	Medical Patients	localhost	Medium
82	dlsadmin	Completed	2012-06-27 16:15...	1	0:0:16	Re-run DSA Job	localhost	Medium
81	dlsadmin	Completed	2012-06-27 16:06...	8	0:0:12	Re-run DSA Job	localhost	Medium
80	dlsadmin	Completed	2012-06-27 16:04...	1	0:0:18	Governance Studi...	localhost	Medium
79	dlsadmin	Completed	2012-06-27 16:04...	1	0:0:0	Complete DSA Job	localhost	Medium

Additionally, the priority parameter that was set is displayed in the Priority column.

The following Oracle DataLens Server Administration Server Job Detail Web page illustrates the operation parameters supplied in the processListRequest operation as in Figure 2.

Figure 2 DSA Job Details and Corresponding Request Parameters



DataLens DSA Job Detail

Job ID 83
Status Completed
Definition patient_care
Description Medical Patients
Start Time 2012-06-28 09:04:13
Finish Time 2012-06-28 09:04:24.097
Duration 0:0:10
Created By dlsadmin
Input Line Count 1530
Output Line Count (GOOD) 1530
Output Line Count (Not Processed) 1530
Output Path/File /u01/datalens_input
Run-Time Locale en_US

Datalens DSA Job Step Detail

Step Name	Type	Status	Description	Start Time	Finish Time	Duration	Input Line Count	Output Line Count	Cor
Text Input	Input	Completed	Text Data Input	2012-06-28 09:04...	2012-06-28 09:04...	0:0:0	0	1530	
Extract_and_Stan...	Processing	Completed	Extract and Stand...	2012-06-28 09:04...	2012-06-28 09:04...	0:0:9	1530	0	
Attributes_and_A...	Output	Completed	Result of attribute...	2012-06-28 09:04...	2012-06-28 09:04...	0:0:0	0	0	
Route_Exceptions	Processing	Completed	Route unrecognize...	2012-06-28 09:04...	2012-06-28 09:04...	0:0:0	1530	1530	
For_Review	Output	Completed	Unrecognized line i...	2012-06-28 09:04...	2012-06-28 09:04...	0:0:0	1530	0	

The following sections provide examples of request and response operations for both the processListRequest and processLineRequest operations:

SOAP Doc-Lit Multi-Line ProcessList Request Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ws="http://www.oracle.com/ws">
<soapenv:Header/>
   <soapenv:Body>
      <ws:processList>
         <dsaName>sampleDSA</dsaName>
         <!--Zero or more repetitions:-->
         <linesOfData>1^res, 17ohm, 19watt, 10%</linesOfData>
         <linesOfData>2^res, 27ohm, 29watt, 20%</linesOfData>
         <linesOfData>3^res, 37ohm, 39watt, 30%</linesOfData>
         <!--Zero or more repetitions:-->
         <dbParameters>?</dbParameters>
         <!--Optional:-->
         <priority>1</priority>
         <!--Optional:-->
         <runtimeLocale>en_US</runtimeLocale>
         <!--Optional:-->
         <fieldSeparator>^</fieldSeparator>
         <!--Optional:-->
         <application>dlsadmin</application>
         <!--Optional:-->
         <description>Example list Doc-Lit client call</description>
      </ws:processList>
   </soapenv:Body>
</soapenv:Envelope>
```

SOAP Doc-Lit Multi-Line `ProcessList` Response Example

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
      <ns2:processListResponse xmlns:ns2="http://www.oracle.com/ws">
         <return>1^Resistor, 17 Ohm, 10%, 19 Watt^32121609^Fixed
resistors^Resistor^Item_Name^RESISTOR^Item_Type^^Resistance^17
OHM^Power^19^Tolerance^10%^Package_Size^^Construction^^Mounting^^Pin_Count
^^For_sale_packaging^</return>
         <return>2^Resistor, 27 Ohm, 20%, 29 Watt^32121609^Fixed
resistors^Resistor^Item_Name^RESISTOR^Item_Type^^Resistance^27
OHM^Power^29^Tolerance^20%^Package_Size^^Construction^^Mounting^^Pin_Count
^^For_sale_packaging^</return>
         <return>3^Resistor, 37 Ohm, 30%, 39 Watt^32121609^Fixed
resistors^Resistor^Item_Name^RESISTOR^Item_Type^^Resistance^37
OHM^Power^39^Tolerance^30%^Package_Size^^Construction^^Mounting^^Pin_Count
^^For_sale_packaging^</return>
      </ns2:processListResponse>
   </S:Body>
</S:Envelope>
```

SOAP Document-Literal One Line Request Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ws="http://www.oracle.com/ws">
<soapenv:Header/>
   <soapenv:Body>
      <ws:processOneLine>
         <dsaName>sampleDSA</dsaName>
         <!--Optional:-->
         <line>1^res, 17ohm, 19watt, 20%</line>
         <!--Zero or more repetitions:-->
         <dbParameters>?</dbParameters>
         <!--Optional:-->
         <priority>1</priority>
         <!--Optional:-->
         <runtimeLocale>en_US</runtimeLocale>
         <!--Optional:-->
         <fieldSeparator>^</fieldSeparator>
         <!--Optional:-->
         <application>dlsadmin</application>
         <!--Optional:-->
         <description>Example Doc-Lit client call</description>
      </ws:processOneLine>
   </soapenv:Body>
</soapenv:Envelope>
```

SOAP Document-Literal One Line Response Example

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
      <ns2:processOneLineResponse xmlns:ns2="http://www.oracle.com/ws">
         <return>1^Resistor, 17 Ohm, 20%, 19 Watt^32121609^Fixed
resistors^Resistor^Item_Name^RESISTOR^Item_Type^^Resistance^17
OHM^Power^19^Tolerance^20%^Package_Size^^Construction^^Mounting^^Pin_Count
^^For_sale_packaging^</return>
      </ns2:processOneLineResponse>
   </S:Body>
</S:Envelope>
```

processDBRequest

This call takes the database parameters as input and returns the DSA Job ID. This call is asynchronous. The parameters are as follows and are the same as described in "processListRequest and processOneLineRequest Operations" on page 1-3:

```
dsaName
dbParameters
priority
runtimeLocale
fieldSeparatorChar
application
description
```

However, the dbParameters parameter is the value that is input to the DSA database query and is related to the DSA job run.

The following sections provide example request and response operations for the processDb operation:

SOAP Document-Literal processDb Request Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ws="http://www.oracle.com/ws">
<soapenv:Header/>
   <soapenv:Body>
      <ws:processDB>
         <dsaName>SampleDSADbInput</dsaName>
         <!--Zero or more repetitions:-->
         <dbParameters>1</dbParameters>
         <dbParameters>2</dbParameters>
         <!--Optional:-->
         <priority>2</priority>
         <!--Optional:-->
         <runtimeLocale>en_US</runtimeLocale>
         <!--Optional:-->
         <fieldSeparator>|</fieldSeparator>
         <!--Optional:-->
         <application>dlsadmin</application>
         <!--Optional:-->
         <description>Example Db Input Doc-Lit client call</description>
      </ws:processDB>
   </soapenv:Body>
</soapenv:Envelope>
```

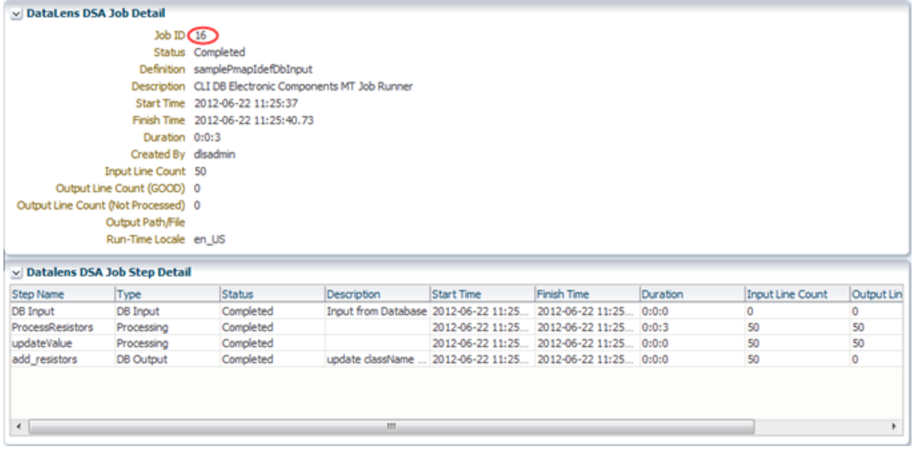
The field separator will be used when the output from the database job is a text file.

SOAP Document-Literal `processDb` Response Example

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
      <ns2:processDBResponse xmlns:ns2="http://www.oracle.com/ws">
         <return>16</return>
      </ns2:processDBResponse>
   </S:Body>
</S:Envelope>
```

The return value of the preceding `processDb` call is the DSA Job ID, in this case 16. This job reads data from the database and updates other fields in the database as in Figure 3.

Figure 3   DSA Job ID Database Process Request



DataLens DSA Job Detail

Job ID	16
Status	Completed
Definition	samplePmapIdefDbInput
Description	CLI DB Electronic Components MT Job Runner
Start Time	2012-06-22 11:25:37
Finish Time	2012-06-22 11:25:40.73
Duration	0:0:3
Created By	dlsadmin
Input Line Count	50
Output Line Count (GOOD)	0
Output Line Count (Not Processed)	0
Output Path/File	
Run-Time Locale	en_US

Datalens DSA Job Step Detail

Step Name	Type	Status	Description	Start Time	Finish Time	Duration	Input Line Count	Output Lin
DB Input	DB Input	Completed	Input from Database	2012-06-22 11:25...	2012-06-22 11:25...	0:0:0	0	0
ProcessResistors	Processing	Completed		2012-06-22 11:25...	2012-06-22 11:25...	0:0:3	50	50
updateValue	Processing	Completed		2012-06-22 11:25...	2012-06-22 11:25...	0:0:0	50	50
add_resistors	DB Output	Completed	update className ...	2012-06-22 11:25...	2012-06-22 11:25...	0:0:0	50	0

Related Documents

For more information, see the following documents in the documentation set:

- The *Oracle Enterprise Data Quality for Product Data Getting Started Guide* provides information about how to get started with EDQP.
- The *Oracle Enterprise Data Quality for Product Data Oracle DataLens Server Installation Guide* provides detailed Oracle DataLens Server installation instructions.
- The *Oracle Enterprise Data Quality for Product Data R12 PIM Connector Installation Guide* provides installation and configuration of Enterprise DQ for Product R12 PIM Connector.
- The *Oracle Enterprise Data Quality for Product Data R12 PIM Connector User's Guide* provides highlights of the core process steps and features of Enterprise DQ for Product R12 PIM Connector.

See the latest version of this and all documents in the Oracle Enterprise Data Quality for Product Data Documentation Web site at http://docs.oracle.com/cd/E35636_01/index.htm

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Oracle Enterprise Data Quality for Product Data Web Service Access to Oracle DataLens Servers Interface Guide, Release 11g R1 (11.1.1.6)
E29146-01