Oracle® Enterprise Data Quality

Automated Package Loading

Release 11g R1 (11.1.1.7)

E40045-02

October 2013

This document provides an introduction to the Oracle Enterprise Data Quality (EDQ) autorun functionality, which allows EDQ to load projects and run jobs when the application server starts up. It explains how the system's autorun functionality is configured, introduces the chore types that can be performed by using the autorun facility and provides an example of an autorun script. The XML schema on which the autorun scripts are based is provided in full.

This document is intended for administrators responsible for configuring the automated execution of jobs. This document assumes that you are familiar with core EDQ concepts and can understand and write XML documents and schemas.

1 The Autorun Facility

EDQ can be configured to perform a range of tasks when the application server starts up. Each task, which is composed of chores, can be configured to be performed every time the application server is started, or just once the next time the application server is started.

Autorun processing is controlled in EDQ by placing autorun scripts that specify tasks in one of two specific directories in the EDQ installation.

1.1 The Configuration Directories

The EDQ `autorun` directory is found in the local configuration directory. It contains two subdirectories named `onceonly` and `startup`. When the application server starts up, EDQ checks the `onceonly` and `startup` directories for autorun scripts and processes any that are present.

Scripts in the `startup` directory are processed every time the EDQ application server starts up. Scripts in the `onceonly` directory are processed when the EDQ application server next starts up, and are then moved to the `complete` subdirectory within `onceonly`. Scripts in the `complete` directory are not processed on subsequent start ups.

1.2 The Autorun Chores

Various kinds of autorun chores are available in EDQ, each with a set of attributes specific to its function. The chore types and their available attributes are defined by the autorun file XML schema, see "The Autorun Schema" on page 3. The chores available are listed in the following table:

ORACLE®

Chore Type	What the Chore Does
httpget	Downloads files from a web server.
package	Loads a project from a .dxi file into the server, or saves a project on the server into a .dxi file. If no nodes are specified then the contents of the whole file, including system level components, are loaded into the server.
runjob	Runs an existing job on the server.
dbscript	Runs a database script against the Director database. This kind of chore must only be used with extreme care, as inappropriately applied scripts may corrupt the underlying database.
sleep	Waits for a specified interval before proceeding.

1.3 The Autorun Scripts

Autorun scripts are files containing XML code. The main part of an autorun script consists of a list of chores, bounded by <chores> tags. Each chore is of one of the autorun chore types listed in "The Autorun Chores" and includes a set of attributes that specify the chore to be performed. The attributes available depend on the chore type selected.

The XML schema used to structure autorun scripts is given in full in "The Autorun Schema" on page 3.

1.3.1 Example

The following XML code shows a sample autorun script. The chores in this script instruct EDQ to:

1. Download the 23People.dxi file, overwriting any existing file with the same name.
2. Import the 23People project from the 23People.dxi file, overwriting any existing project with the same name.
3. Run the 23People Excel.23People job with the rp1 run profile. If this run profile specifies that the job is to run with a run label, it is run as a Server Console job using the runopsjob command and its published results are viewable in Server Console. Otherwise, the job is run as a Director job using the runjob command and its results are viewable only in Director.

```
<?xml version="1.0" encoding="UTF-8"?>
<chores version="1">
<!-- Get the dxi file -->
<httpget overwrite="true" todir="dxiland" tofile="23People.dxi">
<url>http://svn/repos/dev/trunk/benchmark/ benchmark/dxis/23People.dxi</url>
</httpget>
<!-- Import the project from the dxi -->
<package direction="in" dir="dxiland" file="23People.dxi" overwrite="true">
<node type="project" name="23People"/>
</package>
<!-- Run the jobs -->
<runjob project="23People" job="23People Excel.23People" runprofile="rp1"
waitforcompletion="true"/>
</chores>
```

2 The Autorun Schema

This is the Autorun XML schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<!-- Chores -->
<xs:element name="chores">
<xs:complexType>
<!--
List of chores that need to be performed.
The chores will be performed in the order
specified in the xml file
-->
<xs:choice minOccurs="0" maxOccurs="unbounded">
<xs:element name="httpget" type="httpgetType"/>
<xs:element name="package" type="packageType"/>
<xs:element name="runjob" type="runjobType"/>
<xs:element name="dbscript" type="dbScriptType"/>
<xs:element name="sleep" type="sleepType"/>
</xs:choice>
<!-- Schema version number -->
<xs:attribute name="version" type="xs:positiveInteger" use="required"/>
</xs:complexType>
</xs:element>
<!-- Base type for chores -->
<xs:complexType name="choreType">
<!-- Flag indicating whether we should wait for completion
before moving on to the next chore. -->
<xs:attribute name="waitforcompletion" type="xs:boolean" use="optional"
default="true"/>
</xs:complexType>
<!-- HTTP Get chore. Download the specified urls. -->
<xs:complexType name="httpgetType">
<xs:complexContent>
<xs:extension base="choreType">
<xs:sequence minOccurs="1" maxOccurs="1">
<!-- URL to download. -->
<xs:element name="url" type="xs:string"/>
</xs:sequence>
<!-- Filename to download to. -->
<xs:attribute name="tofile" type="xs:string" use="required"/>
<!--
Directory to download the files to.
- relative path is relative to the config dir
- absolute path is used as is
- no path indicates the config dir
-->
<xs:attribute name="todir" type="xs:string" use="optional"/>
<!-- If true existing files are overwritten, otherwise download is not performed.
-->
<xs:attribute name="overwrite" type="xs:boolean" use="optional" default="true"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<!-- dxi file control chore.
Import or export to/from a dxi file. -->
<xs:complexType name="packageType">
<xs:complexContent>
<xs:extension base="choreType">
```

```
<!-- List of root level nodes to import/export. An empty list indicates 'all'. -->
<xs:sequence minOccurs="0" maxOccurs="unbounded">
<xs:element name="node" type="packageNodeType"/>
</xs:sequence>
<!-- dxi filename. -->
<xs:attribute name="file" type="xs:string" use="required"/>
<!--
Directory that the dxi is in.
- relative path is relative to the config dir
- absolute path is used as is
- no path indicates the config dir
-->
<xs:attribute name="dir" type="xs:string" use="optional"/>
<!-- If true existing files/nodes are overwritten, otherwise no operation. -->
<xs:attribute name="overwrite" type="xs:boolean" use="optional" default="true"/>
<!-- Direction: in=import out=export -->
<xs:attribute name="direction" type="packageDirectionEnum" use="required"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<!-- Package node for import or export from/to a dxi. -->
<xs:complexType name="packageNodeType">
<!-- the type of the node to process -->
<xs:attribute name="type" type="nodeTypeEnum" use="required"/>
<!-- the name of the node to process -->
<xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
<!-- db script control chore. Runs db script against the
configuration database. -->
<xs:complexType name="dbScriptType">
<xs:complexContent>
<xs:extension base="choreType">
<!-- db script filename. -->
<xs:attribute name="file" type="xs:string" use="required"/>
<!--
Directory that the db script is in.
- relative path is relative to the config dir
- absolute path is used as is
- no path indicates the config dir
-->
<xs:attribute name="dir" type="xs:string" use="optional"/>
<!-- The database to run the script against -->
<xs:attribute name="database" type="databaseEnum" use="required"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<!-- Invoke named job chore. Run a named job -->
<xs:complexType name="runjobType">
<xs:complexContent>
<xs:extension base="choreType">
<!-- Project name -->
<xs:attribute name="project" type="xs:string" use="required"/>
<!-- Job name -->
<xs:attribute name="job" type="xs:string" use="required"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<!-- Enumeration of databases -->
<xs:simpleType name="databaseEnum">
```

```
<xs:restriction base="xs:string">
<xs:enumeration value="director"/>
<xs:enumeration value="results"/>
</xs:restriction>
</xs:simpleType>
<!-- Enumeration of valid node types -->
<xs:simpleType name="nodeTypeEnum">
<xs:restriction base="xs:string">
<xs:enumeration value="project"/>
<!-- Probably need to do these sometime
<xs:enumeration value="resource"/>
<xs:enumeration value="datastore"/>
-->
</xs:restriction>
</xs:simpleType>
<!-- Enumeration of packaging direction. -->
<xs:simpleType name="packageDirectionEnum">
<xs:restriction base="xs:string">
<xs:enumeration value="in"/>
<xs:enumeration value="out"/>
</xs:restriction>
</xs:simpleType>
<!-- Sleep chore. Wait for a while before doing otherautorun stuff -->
<xs:complexType name="sleepType">
<xs:complexContent>
<xs:extension base="choreType">
<!-- seconds to wait. -->
<xs:attribute name="time" type="xs:integer" use="required"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:schema>
```

3 Related Documents

For more information, see the following documents in the Oracle Enterprise Data Quality documentation set:

- *Oracle Enterprise Data Quality Release Notes*
- *Oracle Enterprise Data Quality Architecture Guide*

See the latest version of this and all documents in the Oracle Enterprise Data Quality Documentation website at

http://download.oracle.com/docs/cd/E48549_01/index.htm

4 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 Automated Package Loading, Release 11g R1 (11.1.1.7)
E40045-02