SailPoint Object Exporter

User Guide

December 2014

Table of Contents

Document Control

Revision History

Version	Author	Reason For Issue	Date
0.1	Paul Wheeler	Initial Draft	15/12/2014
1.0	Paul Wheeler	Minor corrections	22/05/2015

Overview

The Object Exporter is a custom IdentityIQ task for quick and easy export of XML objects such as applications, rules, workflows, configurations and any other objects that are managed in IIQ. The task has the following features:

- Exports XML objects to the same filesystem structure used by the Services Standard Build (SSB)
- Allows the user to specify which classes get exported
- Can optionally export only the objects that have been created or modified after a given date
- Can optionally remove IDs and timestamps from the exported objects
- Allows definition of a naming format for the exported XML files
- Can optionally produce tokenized XML files ready for use in the SSB build, using reverse-lookup to the tokens in a target.properties file.

The tool is designed to help developers working in an IdentityIQ sandbox or development environment, particularly when working directly within IIQ rather than working on external master files and importing those. When working directly within IIQ it can be easy to lose track of what was changed and when – making it difficult to keep the build up-to-date. The Object Exporter provides the following advantages over other export methods:

- Allows the developer to easily export everything they have been working on over a given period, or since the start of a project, removing the need to maintain scripts used to export objects from the console.
- By maintaining a standard naming format for the resulting files it also helps to prevent the creation of multiple copies of the same object XML when there is more than one developer working on an IdentityIQ project.
- Avoids the need to tediously replace text in the XML files with the tokens used in the SSB.
- Avoids the need to create the SSB folder structure when exporting objects.
- The resulting XML files can be added directly to the SSB since they are in the correct folder structure and have the correct tokens.
- The simple process also makes it easy to provide exported files that can be uploaded to a code repository or just stored outside of IIQ as a quick and simple backup of everything that has changed.
- Fixes issues that the console checkout process has with exporting IDs rather than names for objects referenced by certain classes such as IdentityTrigger and Scope.

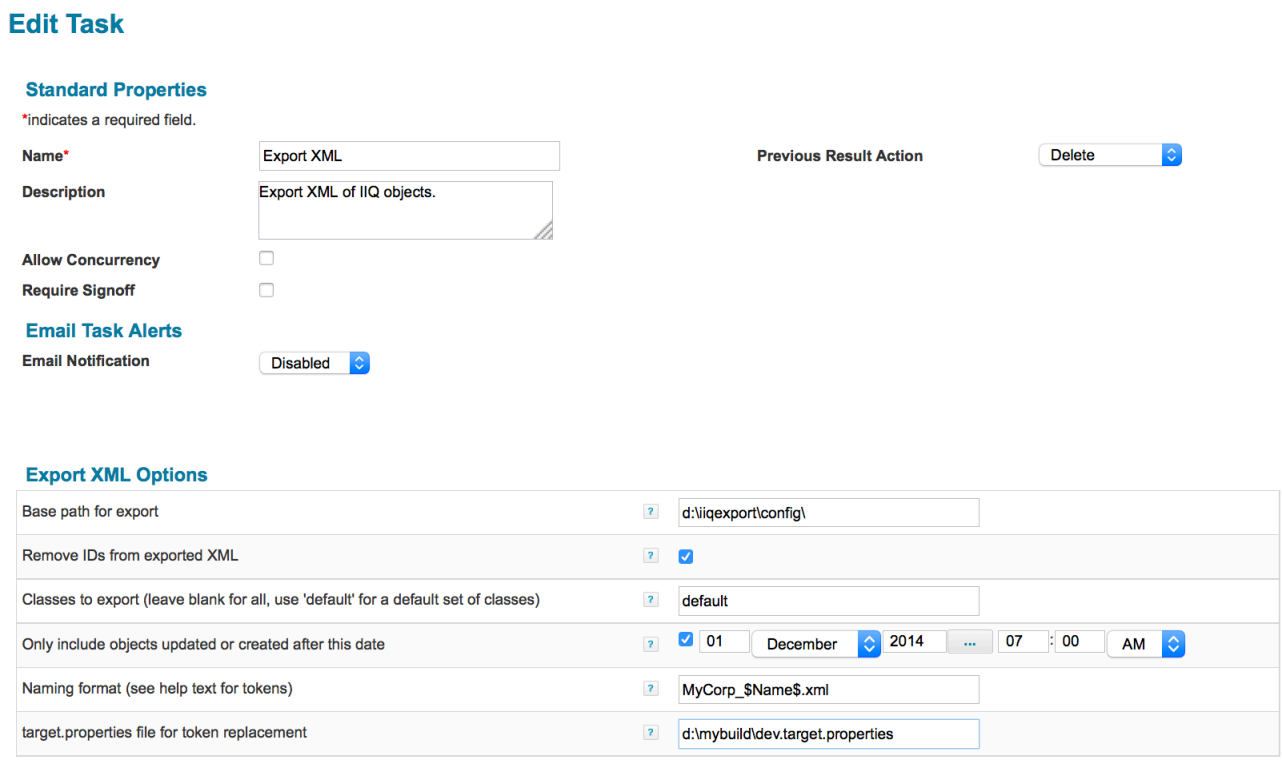
Installation

The Object Exporter consists of a Java class (ExportXML.class) and a Task Definition XML file (TaskDefinition_Export_XML.xml). To install, follow these steps:

- Shutdown the application server
- Add ExportXML.class to WEB-INF/classes/sailpoint/services/task/
- Restart the application server
- Import the Task Definition xml file (using the console or through System Setup -> Import from File).

Using the Object Exporter

To access the Object Exporter in IdentityIQ, go to Monitor -> Tasks. Create a new task of type "XML Object Exporter". The task has several user-configurable options:



Base path for export
Enter the path to which you want to export the XML files. This must be a path that can be accessed by the server that runs the task.

Remove IDs from exported XML
Defines whether the exported XML files will include the IDs and created/modified timestamps for the objects in the environment that you are exporting from. In most cases this should be set to True.

Classes to export
This should be a comma-separated lists of object types to export. If left blank it will export all object types. If set to "default" it will export only the types of objects that are commonly exported during an IdentityIQ project. It can also be a combination of "default" and other classes – for example you could use "default,Scope,Bundle".

Only include objects updated or created after this date
If enabled, this option allows the user to enter a date and time which defines the earliest creation and modification timestamps by which we will filter the objects to be included in the export. If disabled, creation and modification timestamps will be ignored.

Naming format
Allows the user to define a format for naming the exported XML files. There are two optional tokens that can be used:

- $Name$ The name of the XML object
- $Class$ The name of the object class, e.g. Application, Rule etc.

Using these tokens you can build a custom naming format for the exported files. For example, "MyCorp_$Class$_$Name$.xml" would export an application called "ActiveDirectory" with the file name "MyCorp_Application_ActiveDirectory.xml". If this option is left blank the XML file will be named as the object name with a ".xml" extension.

Illegal characters and spaces will be replaced with underscores in the file names.

target.properties file for token replacement
Optionally defines the path to a target.properties file that provides token values for the sandbox/development environment in the usual SSB format. The task performs a "reverse-lookup" on any text it finds in the XML that matches a token value found in the target.properties file, and replaces it with the corresponding token. If this option is left blank, no tokenization of the export files will take place.

After selecting the desired options, click "Save and Execute". The task will start running and its progress can be monitored in the Task Results. When completed, the results will show a summary of the number of XML files exported for each class and a total of all objects exported.