OAI

OAI Interfaces

OAI-PMH Server

In the following sections and subpages, you will learn how to configure OAI-PMH server and activate additional OAI-PMH crosswalks. The user is also referred to OAI-PMH Data Provider for greater depth details of the program.

The OAI-PMH Interface may be used by other systems to harvest metadata records from your DSpace.

OAI-PMH Server Activation

To enable DSpace's OAI-PMH server, just make sure the `[dspace]/webapps/oai/` web application is available from your Servlet Container (usually Tomcat).

- You can test that it is working by sending a request to: `http://[full-URL-to-OAI-PMH]/request?verb=Identify`
- The response should look similar to the response from the DSpace Demo Server: http://demo.dspace.org/oai/request?verb=Identify

If you're using a recent browser, you should see a HTML page describing your repository. What you're getting from the server is in fact an XML file with a link to an XSLT stylesheet that renders this HTML in your browser (client-side). Any browser that cannot interpret XSLT will display pure XML. The default stylesheet is located in `[dspace]/webapps/oai/static/style.xsl` and can be changed by configuring the `stylesheet` attribute of the `Configuration` element in `[dspace]/config/crosswalks/oai/xoai.xml`.

Relevant Links

- OAI 2.0 Server - basic information needed to configure and use the OAI Server in DSpace
- OAI-PMH Data Provider 2.0 (Internals) - information on how it's implemented
- http://www.openarchives.org/pmh/ - information on the OAI-PMH protocol and its usage (not DSpace-specific)

OAI-PMH Server Maintenance

After activating the OAI-PMH server, you need to also ensure its index is updated on a regular basis. Currently, this doesn't happen automatically within DSpace. Instead, you must schedule the `[dspace.dir]/bin/dspace oai import` commandline tool to run on a regular basis (usually at least nightly, but you could schedule it more frequently).

Here's an example cron that can be used to schedule an OAI-PMH reindex on a nightly basis (for a full list of recommended DSpace cron tasks see Scheduled Tasks via Cron):

```
# Update the OAI-PMH index with the newest content (and re-optimize that index) at midnight every day
# NOTE: ONLY NECESSARY IF YOU ARE RUNNING OAI-PMH
# (This ensures new content is available via OAI-PMH and ensures the OAI-PMH index is optimized for better performance)
0 0 * * * [dspace.dir]/bin/dspace oai import -o > /dev/null
```

More information about the `dspace oai` commandline tool can be found in the OAI Manager documentation.

OAI-PMH / OAI-ORE Harvester (Client)

This section describes the parameters used in configuring the OAI-ORE / OAI-ORE harvester (for XMLUI only). This harvester can be used to harvest content (bitstreams and metadata) into DSpace from an external OAI-PMH or OAI-ORE server.

Relevant Links

For information on activating & using the OAI-PMH / OAI-ORE Harvester to harvest content into your DSpace, see Harvesting Items from XMLUI via OAI-ORE or OAI-PMH

Harvesting from another DSpace

If you are harvesting content (bitstreams and metadata) **from** an external DSpace installation via OAI-PMH & OAI-ORE, you first should verify that the external DSpace installation allows for OAI-ORE harvesting.

First, that external DSpace must be running both the OAI-PMH interface and the XMLUI interface to support harvesting content from it via OAI-ORE.

You can verify that OAI-ORE harvesting option is enabled by following these steps:

1. First, check to see if the external DSpace reports that it will support harvesting ORE via the OAI-PMH interface. Send the following request to the DSpace's OAI-PMH interface: `http://[full-URL-to-OAI-PMH]/request?verb=ListRecords&metadataPrefix=ore`
    - The response should be an XML document containing ORE, similar to the response from the DSpace Demo Server: http://demo.dspace.org/oai/request?verb=ListRecords&metadataPrefix=ore
2. Next, you can verify that the XMLUI interface supports OAI-ORE (it should, as long as it's a current version of DSpace). First, find a valid Item Handle. Then, send the following request to the DSpace's XMLUI interface: `http://[full-URL-to-XMLUI]/metadata/handle/[item-handle]/ore.xml`
    - The response should be an OAI-ORE (XML) document which describes that specific Item. It should look similar to the response from the DSpace Demo Server: http://demo.dspace.org/xmlui/metadata/handle/10673/3/ore.xml

OAI-PMH / OAI-ORE Harvester Configuration

There are many possible configuration options for the OAI harvester. Most of these are contained in the `[dspace]/config/modules/oai.cfg` file (unless otherwise noted below). They may be updated there or overridden in your `local.cfg` config file (see Configuration Reference).

Configuration File:	**`[dspace]/config/modules/oai.cfg`**
Property:	`oai.harvester.eperson`
Example Value:	`oai.harvester.eperson = admin@myu.edu`
Informational Note:	The EPerson under whose authorization automatic harvesting will be performed. This field does not have a default value and must be specified in order to use the harvest scheduling system. This will most likely be the DSpace admin account created during installation.
Property:	`oai.url`
Example Value:	`oai.url = ${dspace.baseUrl}/oai`
Informational Note:	The base url of the OAI-PMH disseminator webapp (i.e. do not include the /request on the end). This is necessary in order to mint URIs for ORE Resource Maps. The default value of `${dspace.baseUrl}/oai` will work for a typical installation, but should be changed if appropriate. Please note that `dspace.baseUrl` is defined in your `dspace.cfg` configuration file.
Property:	`oai.ore.authoritative.source`
Example Value:	`oai.ore.authoritative.source = oai | xmlui`
Informational Note:	The webapp responsible for minting the URIs for ORE Resource Maps. If using oai, the `oai.url` config value must be set. • When set to 'oai', all URIs in ORE Resource Maps will be relative to the OAI-PMH URL (configured by `oai.url` above) • When set to 'xmlui', all URIs in ORE Resource Maps will be relative to the DSpace Base URL (configued by `dspace.url` in the `dspace.cfg` file) The URIs generated for ORE ReMs follow the following convention for either setting: http://\[base-URL\]/metadata/handle/\[item-handle\]/ore.xml
Property:	`oai.harvester.autoStart`
Example Value:	`oai.harvester.autoStart = false`
Informational Note:	Determines whether the harvest scheduler process starts up automatically when the XMLUI webapp is redeployed.
Property:	`oai.harvester.metadataformats.PluginName`
Example Value:	`oai.harvester.metadataformats.PluginName = \` `http://www.openarchives.org/OAI/2.0/oai_dc/, Simple Dublin Core`
Informational Note:	This field can be repeated and serves as a link between the metadata formats supported by the local repository and those supported by the remote OAI-PMH provider. It follows the form `oai.harvester.metadataformats.PluginName = NamespaceURI,Optional Display Name` . The pluginName designates the metadata schemas that the harvester "knows" the local DSpace repository can support. Consequently, the PluginName must correspond to a previously declared ingestion crosswalk. The namespace value is used during negotiation with the remote OAI-PMH provider, matching it against a list returned by the ListMetadataFormats request, and resolving it to whatever metadataPrefix the remote provider has assigned to that namespace. Finally, the optional display name is the string that will be displayed to the user when setting up a collection for harvesting. If omitted, the PluginName:NamespaceURI combo will be displayed instead.

Property:	`oai.harvester.oreSerializationFormat.OREPrefix`
Example Value:	`oai.harvester.oreSerializationFormat.OREPrefix = \` `http://www.w3.org/2005/Atom`
Informational Note:	This field works in much the same way as `oai.harvester.metadataformats.PluginName` . The `OREPrefix` must correspond to a declared ingestion crosswalk, while the Namespace must be supported by the target OAI-PMH provider when harvesting content.
Property:	`oai.harvester.timePadding`
Example Value:	`oai.harvester.timePadding = 120`
Informational Note:	Amount of time subtracted from the from argument of the PMH request to account for the time taken to negotiate a connection. Measured in seconds. Default value is 120.
Property:	`oai.harvester.harvestFrequency`
Example Value:	`oai.harvester.harvestFrequency = 720`
Informational Note:	How frequently the harvest scheduler checks the remote provider for updates. Should always be longer than *timePadding* . Measured in minutes. Default value is 720.
Property:	`oai.harvester.minHeartbeat`
Example Value:	`oai.harvester.minHeartbeat = 30`
Informational Note:	The heartbeat is the frequency at which the harvest scheduler queries the local database to determine if any collections are due for a harvest cycle (based on the *harvestFrequency*) value. The scheduler is optimized to then sleep until the next collection is actually ready to be harvested. The *minHeartbeat* and *maxHeartbeat* are the lower and upper bounds on this timeframe. Measured in seconds. Default value is 30.
Property:	`oai.harvester.maxHeartbeat`
Example Value:	`oai.harvester.maxHeartbeat = 3600`
Informational Note:	The heartbeat is the frequency at which the harvest scheduler queries the local database to determine if any collections are due for a harvest cycle (based on the *harvestFrequency*) value. The scheduler is optimized to then sleep until the next collection is actually ready to be harvested. The *minHeartbeat* and *maxHeartbeat* are the lower and upper bounds on this timeframe. Measured in seconds. Default value is 3600 (1 hour).
Property:	`oai.harvester.maxThreads`
Example Value:	`oai.harvester.maxThreads = 3`
Informational Note:	How many harvest process threads the scheduler can spool up at once. Default value is 3.
Property:	`oai.harvester.threadTimeout`
Example Value:	`oai.harvester.threadTimeout = 24`
Informational Note:	How much time passes before a harvest thread is terminated. The termination process waits for the current item to complete ingest and saves progress made up to that point. Measured in hours. Default value is 24.
Property:	`oai.harvester.unknownField`
Example Value:	`oai.harvester.unkownField = fail | add | ignore`
Informational Note:	You have three (3) choices. When a harvest process completes for a single item and it has been passed through ingestion crosswalks for ORE and its chosen descriptive metadata format, it might end up with DIM values that have not been defined in the local repository. This setting determines what should be done in the case where those DIM values belong to an already declared schema. *Fail* will terminate the harvesting task and generate an error. **Ignore** will quietly omit the unknown fields. **Add** will add the missing field to the local repository's metadata registry. Default value: **fail**.
Property:	`oai.harvester.unknownSchema`
Example Value:	`oai.harvester.unknownSchema = fail | add | ignore`

Informational Note:	When a harvest process completes for a single item and it has been passed through ingestion crosswalks for ORE and its chosen descriptive metadata format, it might end up with DIM values that have not been defined in the local repository. This setting determines what should be done in the case where those DIM values belong to an unknown schema. **Fail** will terminate the harvesting task and generate an error. **Ignore** will quietly omit the unknown fields. **Add** will add the missing schema to the local repository's metadata registry, using the schema name as the prefix and "unknown" as the namespace. Default value: **fail**.
Property:	`oai.harvester.acceptedHandleServer`
Example Value:	`oai.harvester.acceptedHandleServer = \` `hdl.handle.net, handle.test.edu`
Informational Note:	A harvest process will attempt to scan the metadata of the incoming items (identifier.uri field, to be exact) to see if it looks like a handle. If so, it matches the pattern against the values of this parameter. If there is a match the new item is assigned the handle from the metadata value instead of minting a new one. Default value: *hdl.handle.net* .
Property:	`oai.harvester.rejectedHandlePrefix`
Example Value:	`oai.harvester.rejectedHandlePrefix = 123456789, myeduHandle`
Informational Note:	Pattern to reject as an invalid handle prefix (known test string, for example) when attempting to find the handle of harvested items. If there is a match with this config parameter, a new handle will be minted instead. Default value: *123456789* .