Creating Custom Collection Objects

In certain cases, you may wish to change the nature or behaviour of Collection Objects in Islandora. By creating a custom collection object, you can override the default behaviour of Islandora. A simplified overview of Collection Objects is provided in the introduction of this guide. For example, you can return objects that have a different relationship to the collection object, you can present objects in your collection in a custom way to your viewer, and you can create security policies that will restrict access to the items in your collection (overriding Fedora's default behaviour). The following chapter provides more information about the default behavior of Islandora, how Collection Objects can be constructed, and how they can be extended and customized.

Overview:

Collection Objects have one mandatory Datastream (COLLECTION_POLICY) and several optional Datastreams. The optional Datastreams override the default behaviour of the Islandora Module. You may add Datastreams by navigating to the collection object you wish to modify, and then adding Datastreams via the interface. You may also add these Datastreams using any Fedora tools that you are familiar with.

- COLLECTION_POLICY*
- QUERY
- COLLECTION_VIEW
- CHILD_SECURITY: gives a POLICY Datastream to child objects

**Mandatory*

COLLECTION_POLICY

A Collection Object can have four Datastreams, although the COLLECTION_POLICY is the only mandatory stream. If you do not have a COLLECTION_POLICY Datastream, additional objects cannot be ingested as members of that collection object. In other words, in order to add items to a collection or sub-collection, the collection object (or **"parent-type" object**) must have a COLLECTION_POLICY stream. Here is an example of a COLLECTION_POLICY Datastream (as viewed using the Islandora interface to view in a browser)

```
– <collection_policy name="" xsi:schemaLocation="http://www.islandora.ca http://syn.lib.umanitoba.ca
  /collection_policy.xsd">
   – <content_models>
       <content_model dsid="ISLANDORACM" name="Collection" namespace="islandora:collection"
       pid="islandora:collectionCModel"/>
     </content_models>
     <relationship>isMemberOfCollection</relationship>
  </collection_policy>
```

The COLLECTION_POLICY Datastream must have a isMemberOfCollection relationship declared, and must be affiliated with the islandora:collectionCModel.

The relationship statement tells Islandora that this Fedora object is a collection object. Islandora can then query the resource index for objects that have a relationship of isMemberOfCollection to this collection object.

The isMemberOfCollection is the default relationship used by Islandora, but other relationships can be used by declaring that relationship in the COLLECTION_POLICY Datastream. If you use another relationship other than this relationship, you will have to use a QUERY Datastream as well. (In other words, any new relationship declared in the COLLECTION_POLICY Datastream will make the QUERY Datastream mandatory.)

If you wish to create a new COLLECTION_POLICY stream, you will be writing XML. One way to do this is to start with an example collection policy (there is one available in....) and edit it. The **DSID** of this datastream must be COLLECTION_POLICY.

QUERY

A QUERY Datastream is an ITQL query that overrides the Islandora's default ITQL query. If you have declared different relationships (not a hasModel relationship) in your COLLECTION_POLICY Datastream, you will have to write a custom QUERY stream to return these relationships. In order to do this, you will have to have an understanding of ITQL. Resources for learning ITQL are offered in the Bibliography for this guide. Your ITQL query must return SPARQL XML to be parsed by the default collection view xslt file, or by a custom COLLECTION_VIEW xslt that you have written yourself.

When you write a QUERY Datastream, you ask the Islandora module to retrieve items that have a different set of objects related to your collection object from those in the default ITQL query. The default ITQL query is located in the islandora module in the collection_class.inc file. This is the query:

```
$query_string = 'select $object $title $content from <#ri> where ($object <dc:title>
$title and $object <fedora-model:hasModel>
$content and ($object <fedora-rels-ext:isMemberOfCollection> <info:fedora/' . $pid . '>

or $object <fedora-rels-ext:isMemberOf> <info:fedora/' . $pid . '>)and $object <fedora-
model:state>
<info:fedora/fedora-system:def/model#Active>)minus $content <mulgara:is> <info:fedora
/fedora-system:FedoraObject-3.0>order by $title';
```

Note that if you write a QUERY Datastream, you may also have to write a COLLECTION_VIEW Datastream to parse and display your results. Sample QUERY Datastreams are provided in the Resources section of this guide.

COLLECTION_VIEW

A COLLECTION_VIEW Datastream contains an XSLT that will define how objects in a collection are displayed. You may wish to write a custom COLLECTION_VIEW stream to change the look and feel of your collection for visitors. For a custom XLST used for a COLLECTION_VIEW Datastream, please refer to the samples and resources section. The XSLT in your COLLECTION_VIEW Datastream has to be matched to either the default ITQL query used by Islandora (and found in the Islandora module under sparql_2_html.xsl) or the custom QUERY Datastream that you have written. Your XSLT will parse the SPARQL XML returned by either the default query, or the query you have written. This is the default xslt, called from the Islandora module at object_helper.inc.

CHILD_SECURITY

The optional CHILD_SECURITY Datastream is a hand-written eXtensible Access Control Markup Language (XACML) policy that provides security at the collection level. To learn more about XACML, visit our resources section. The CHILD_SECURITY Datastream interacts with the default set-up of your Fedora repository. In order to use the CHILD_SECURITY stream effectively, you may wish to review the Islandora and Security section of this guide.

The CHILD_SECURITY Datastream overrides whatever default security you have configured as part of your Fedora and Drupal installations (see the Fedora installation section of this document, particularly information about global XACML policies). For example, if objects in your Fedora repository are, by default, available to the public, you may wish to write a CHILD_SECURITY stream for a collection to restrict access to that collection to specific users, or specific **Drupal Roles**.

All of the objects that are ingested as members of a collection object that has a CHILD_SECURITY stream will have a POLICY stream. Without the POLICY Datastream, the objects default to your base security configuration. This means that if you add a CHILD_SECURITY stream to an object after items are already affiliated with the collection, these objects will not adopt the CHILD_SECURITY policies (and they will have no POLICY Datastreams).

Note that Islandora does not change the UI in the case where a POLICY Datastream exists. This means that the icons for managing objects (such as the purge option) will still be available to users. However, if users attempt to perform the action and they do not have permissions corresponding to that action, they will receive an error. We are hoping that future versions of Islandora will not have this limitation.

Generating XACML Policies

XACML Editor

Non-developers may want to use the XACML Editor module to generate XACML policies using a graphical user interface. Further instructions for this module are found in Chapter 5: Using the XACML Editor

User-Generated XACML

Hand-written XACML policy files can be added to the $FEDORA_HOME/data/ fedora-xacml-policies/repository-policies. You can retrieve an example XACML policy file from the Resources section of the guide. However, this example opens API-M to all of the users in your Drupal instance that are authenticated users.

When you write a CHILD_SECURITY stream you are writing a XACML policy. That XACML policy must be parseable (usable) by Islandora's simple parser. Islandora's simple parser expects the CHILD_SECURITY Datastream to contain a XACML policy that denies access to all users, and then provides exceptions for users with certain **Drupal Roles**, or User IDs. If users have IDs or roles that are permitted access in the XACML policy, they will be allowed to ingest, view, or modify elements in that collection. You can view an annotated sample XACML policy in the Resources section of this document. This document can act as a starting point for a collection-object CHILD_SECURITY Datastream.

In order for Islandora to be able to browse collections, your collection object must also have a hasModel entry in the RELS-EXT Datastream that points to islandora:collectionCModel. This lets the module know that the object represents a collection and it will then query for objects that are members of this collection.