WK2D: Advanced Islandora Workshop

June 9, 2014
Helsinki, Finland

Mark Leggott, Islandora Foundation
Erin Tripp, discoverygarden inc
Paul Pound, University of Prince Edward Island
Mark Jordan, Simon Fraser University
Islandora

Agenda

- Introductions
- System & community overview
- Latest Release – Highlights
- Introduction to Solr
- Introduction to Form Builder
Open Source Software Community

http://islandora.ca/islandora-installations
Estimated 150+ sites worldwide
Online Discussion Forums

Welcome to the Islandora Users group. Learn more about Islandora at islandora.ca. If you are a developer, you may want to join our developers group (islandora-dev)

- Islandora Committers Call - Feb 27th, 2014 (1)
  By Melissa Anez - 1 post - 4 views

- jQuery 1.8 and Islandora (4)
  By Everson Kennedy - 4 posts - 11 views

- GitHub Repo roundup (4)
  By Melissa Anez - 4 posts - 17 views

- Islandora Scholar errors (5)
  By Evan Jackson - 5 posts - 17 views

- Video Solution Pack: MP4 creation failed. (1)
  By Janis T - 1 post - 5 views

- Islandora Solr Display Profiles (3)
  By Evan Jackson - 3 posts - 9 views

- Islandora Solr question (3)
  By Evan Jackson - 3 posts - 27 views

- store on islandora (4)
  By Mehdi Haji karimi - 4 posts - 23 views

- creating newspapers with Tuque (9)
  By aaronC - 9 posts - 26 views

https://groups.google.com/forum/#!forum/islandora
Code Contributors on Github

Islandora

islandora_solution_pack_large_image

Islandora module to support ingesting and viewing TIF/TIFF files
Updated an hour ago

islandora_bookmark

Allows creation and management of lists of PIDs that are attached to a Drupal user, and can be shared with other users. These lists can be used to track and sort PIDs, or to export lists of citations. The module can also be used in conjunction with Soir search results to perform some export tasks.
Updated 2 hours ago

https://github.com/Islandora
Open Source Software Stack

- Web interface (Drupal)
- Search (Solr)
- Preservation (Fedora)
Islandora 7.x-1.3

- **SCHOLAR**
  - Embargo
  - Citation Style Language
  - RoMEO Integration
  - Citation Import
  - Citation Export
  - Bookmarks
  - Open Archives Initiative
  - Google Scholar
  - Zotero
  - Bibutils

- **PREMIS**
- **XQUERY**
Solr Overview

- Indexing
- Foxml to Solr
- Gsearch
- Solr field types
- Solr Configuration
- Request Handler
- Solr client configuration
- Metadata display
- Solr facet pages
- Documentation
UPEI Robertson Library setup

- Many separate Drupal installs accessing the same solr instance.
- Search results segmented by PID namespace
- Each site has its own configuration for advanced search
- Each site has their own facets.
What gets indexed?

With Islandora what gets indexed is up to you. If you follow the Islandora instructions you will get MODS, DC, RELS-EXT, Fulltext and more by default.

If you have objects with other text or xml datastreams they can be indexed as well

- If you create a form in Formbuilder you can add an xslst to the Gsearch configuration to index your new metadata. Form builder uses xpath and Gsearch uses xslts so there are many similarities
How does it get indexed?

- Islandora relies on FedoraGsearch to index Fedora objects in Solr. Gsearch is a Java webapp that crosswalks a Fedora FOXML file to a Solr document and sends the result to Solr.
- When a Fedora Object is ingested or modified it sends messages via JMS. Gsearch receives these messages and updates the solr index.

[Link to Raw Solr Result](#)
Foxml To Solr

Gsearch xslt
<xsl:for-each select="mods:abstract">
  <field>
    <xsl:attribute name="name">
      <xsl:value-of select="concat(\$prefix, local-name(), \$suffix)"/>
    </xsl:attribute>
    <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

Solr Document
<doc>
  <field name="PID">islandora:1</field>
  <field name="mods_abstract_ms">
    The text from the mods:abstract element
  </field>
</doc>
Gsearch configuration

Gsearch configs located:

- /var/lib/tomcat6/webapps/fedoragsearch/WEB-INF/classes/fgsconfigFinal

Most interesting files are:

- fedoragearch.properties
- index/FgsIndex/index.properties
- index/FgsIndex/foxmlToSolr.xslt
  - index/FgsIndex/islandora_transforms
- updater/FgsIndex/updater.properties
Gsearch Cont.

- Repository/FgsRepos/repository.properties
- Can use information in datastreams to pull in content from other objects (parent title)
- Gsearch indexing interface
- Rebuilding an entire index using Gsearch interface will not remove objects from the index (update/add).
  - Gsearch will remove entries when configured to listen for events.
Solr field types

Solr can be configured to support different types of fields and these fields can be single or multivalued.

- Single Valued can be used for sorting, multivalued can’t.
- Analyzed (tokenized etc.) fields are good for searching but not so good for display.
Solr Field types cont’d

- In general String (non analyzed) are used for display (facets or results) and

- Text (analyzed) type fields are used for searching. There are also date, numeric and other field types and you can also create your own field types.
Solr configs

Solr configs located:

- /usr/share/solr/core1/conf
  Most interesting files
- solrconfig.xml
  - request handlers (if you want filters that Islandora can’t override)
- schema.xml
  - fields
  - fieldType
  - copyFields
Request Handler

Many of the settings you can set in the Islandora configuration can also be set in a Solr requestHandler.

You can set values in the requestHandler that cannot be overridden by the Islandora module.
Solr Request Handler

```xml
<requestHandler name="guardian" class="solr.SearchHandler">
<!-- default values for query parameters -->
<lst name="defaults">
<str name="echoParams">explicit</str>
<str name="fl">title,subject,genre,form,surveyor,contributor,type,country,county,region,
  city,PID,yearPublished,mods.title_facet,mods.dateissued,mods.pageNum</str>
<str name="q.alt">*:*</str>
<str name="qf"> mods.title^2.5 mods.subject^1.5 mods.surveyor^5.0 mods.contributor^5.0
  mods.country^1.5 mods.county^1.5 mods.region^3.8 mods.city^5.8 mods.genre^2
  mods.form^1.5 PID^0.5 OCR.OCR^3.0 </str>
</lst>
<lst name="appends">
<str name="fq">PID:newspapers* OR PID:guardian* OR PID:upeia\:cadre*</str>
</lst>
</requestHandler>
```
Solr

Fielded search (Islandora advanced search block) vs dismax (Islandora simple search block)

Solr interface
Islandora Solr Client

- The Islandora Solr client can be used as a frontend to any Solr index not just a Fedora based index.
- You would need to provide a custom Display profile for non Fedora content.
- This also allows us to index both Drupal and Fedora content in one index.
Configuring the Solr client

Topics to Cover:

- Blocks ([Link](admin/structure/block))
  - Islandora Simple Search
  - Islandora Displays
  - Islandora Facets
  - Islandora Queries
  - Islandora Sort
Configuration cont’d

- Main Configuration Page
  - url and port
  - request handler

- Solr Settings tab
  - Display profiles
  - Secondary profiles
  - [Link 1](#), [Link 2](#)
Configuration cont’d

- Default Display settings
  - Add and configure fields (permissions etc.)

<table>
<thead>
<tr>
<th>SOLR FIELD</th>
<th>SETTINGS</th>
<th>OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>mods_titleinfo_title_ms</td>
<td>Label: Article Title</td>
<td>configure remove</td>
</tr>
<tr>
<td>dc.contributor</td>
<td>Label: Author(s)</td>
<td>configure remove</td>
</tr>
</tbody>
</table>
Configuration cont’d

- Sort Settings
  - add a field (must be single valued try `fgs_createdDate_dt`)
  - configure and save field

<table>
<thead>
<tr>
<th>SOLR FIELD</th>
<th>SETTINGS</th>
<th>OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fgs_createdDate_dt</code></td>
<td>Label: Date Created</td>
<td>configure  remove</td>
</tr>
<tr>
<td><code>fgs_label_s</code></td>
<td>Label: Article Title</td>
<td>configure  remove</td>
</tr>
</tbody>
</table>
Configuration cont’d

- Facet Settings
  - add a field (Use a String _s try RELS_EXT_hasModel_uri_ms)
  - configure and save field
Configuration cont’d

- Advanced Search
  - add a field (Use a text field try dc.title and dc.description)
  - configure and save fields
  - enable and configure the advanced search block
Configuration cont’d

- Query Defaults
  - namespaces
  - Solr default query (for empty search)
  - Sort field for default query (single valued)
  - Solr base filter
    - Additional filters beyond namespace (you could limit to a certain cmodel or language try RELS_EXT_hasModel_uri_ms: info\:fedora/islandora\:newspaperIssueCModel )
  - Query Fields (these are fields searched by simple search, defType=dismax)
Configuration cont’d

- Required Solr fields
  These are fields that the code that generates the search results depend on.

```
Content model Solr field
RELS_EXT_hasModel_url_ms
Solr field containing the content model URIs. This should be a multivalued string field.

Datastream ID Solr field
fedora_datastreams_ms
Solr field containing the populated datastream IDs. This should be a multivalued string field. If this field is not populated, the DSID of TN will be assumed valid for thumbnails.
```
Metadata Display

Instead of reading a datastream like DC for the Details display (metadata) you can use Solr fields for this display. Link

You need to have the Islandora Metadata display enabled and then choose to use it in the admin/islandora/metadata config section
Metadata display

• Then configure the module here
Islandora Solr Facet Pages

An A to Z list based on a Solr field

Example
Admin interface
Documentation

- Islandora wiki
- Islandora github repository
- Islandora Solr module
- Islandora Solr Metadata display
Form Builder Overview

• Introducing the XML Form Builder

• Solution Pack metadata forms

• Modifying existing forms

• Creating new forms

• Cookbook
What is the XML Form Builder?

- A collection of Drupal modules that allow for the manipulation of XML documents (datastreams) though Drupal forms.
Features

• Works with most metadata standards (that have an XML schema)
• Will work with custom XML schemas
• Highly configurable
• Forms are portable
• Secure
• Built upon the existing Drupal Form API
Drupal Forms API
<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>Nice and pretty xml</td>
<td>6 months ago</td>
</tr>
<tr>
<td>DarwinCore</td>
<td>A reorganization of the existing forms and additions</td>
<td>9 months ago</td>
</tr>
<tr>
<td>EAC-CPF</td>
<td>A reorganization of the existing forms and additions</td>
<td>9 months ago</td>
</tr>
<tr>
<td>FGDC</td>
<td>A reorganization of the existing forms and additions</td>
<td>9 months ago</td>
</tr>
<tr>
<td>LIDO</td>
<td>Nice and pretty xml</td>
<td>6 months ago</td>
</tr>
<tr>
<td>MADS</td>
<td>Nice and pretty xml</td>
<td>6 months ago</td>
</tr>
<tr>
<td>METSRights</td>
<td>Nice and pretty xml</td>
<td>6 months ago</td>
</tr>
<tr>
<td>MODS</td>
<td>Create Caltech MODS</td>
<td>3 months ago</td>
</tr>
<tr>
<td>PBCore</td>
<td>Nice and pretty xml</td>
<td>6 months ago</td>
</tr>
<tr>
<td>README.md</td>
<td>Fixed text and links in README</td>
<td>9 months ago</td>
</tr>
</tbody>
</table>
But...

• It’s not very user friendly
• It’s not very simple
• It’s not very intuitive
• It has some limitations
Form Elements

Supported
• button
• textfield
• textarea
• select
• hidden
• value
• markup
• feldset
• datepicker

Not fully supported
• checkbox
• checkboxes
• date
• file
• managed_file
• password_confirm
• radio
• radios
• tableselect
• vertical_tabs
• weight
• button
• image_button
• submit
# Solution Pack Metadata Forms

<table>
<thead>
<tr>
<th>Title</th>
<th>Type</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Basic image MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Citation MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Compound Object MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Islandora Book MODS Form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Large image MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Newspaper</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Newspaper Issue</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>PDF MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Thesis MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Video MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
<tr>
<td>Web ARCHive MODS form</td>
<td>Built-in</td>
<td>Copy, View, Export, Associate</td>
</tr>
</tbody>
</table>
Book Title

Sub Title

Name

Type
personal

Role
marcrelator
Add

Genre
marcgt

Table of Contents
Title *

Sub Title

Name

Type

- corporate

Name

Role

Select a role from this vocabulary - http://id.loc.gov/vocabulary/relators.html - e.g. Artist, Creator, Designer, Engraver, Illustrator, Photographer, Printmaker, etc.

Add

Type of Resource

- mixed material

Genre

Website

ORIGIN INFORMATION

Date Captured
Using the Form Builder

• Relies on XPath expressions to identify the MODS XML elements (for example) that each form element reads/modifies

• Provides a user interface for
  • browsing elements in schema
  • Assigning common attributes
  • Defining XPath expressions for CRUD (Create/Read/Update/Delete) operations
  • Defining “Advanced Form Controls” and “More Advanced Controls”
XPath

• Language used to identify (or “select”) paths to nodes in XML documents

• Some examples

<table>
<thead>
<tr>
<th>Expression</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>author</td>
<td>Selects all &lt;author&gt; elements in the current context</td>
</tr>
<tr>
<td>//author</td>
<td>Selects all &lt;author&gt; elements in the document</td>
</tr>
<tr>
<td>/books</td>
<td>Selects the top-level element in a document with outer tags</td>
</tr>
<tr>
<td></td>
<td>&lt;books&gt;&lt;/books&gt;</td>
</tr>
<tr>
<td>book/@cost</td>
<td>Selects the value of the “cost” attribute in &lt;book cost=&quot;23.95&quot;&gt;</td>
</tr>
</tbody>
</table>
XPath in the Form Builder interface

• XPaths describe where the action will take place in the XML Document.

• self::node() is common and used in various contexts, most often in Create, Update, and Delete settings

• Explicit path to the element (e.g. “mods:OriginInfo”), most often in Read settings
CRUD Operations

- Create
  - After the form is submitted, the new element/attribute is created and appended to the parent XML element
- Read
  - Before form is displayed, populates it with data from an XML document
- Update
  - After the form is submitted, the value in the form field to is used to update the element/attribute selected
- Delete
  - After the form is submitted, the selected element/attribute is deleted
Create / Update Variables

- Attribute
  - e.g., `<name type="personal">Joe</name>`
- Element
  - e.g., `<genre>book</genre>`
- XML snippets and the use of %value%
  - e.g., `<typeOfResource collection="yes">%value%</typeOfResource>`
Cookbook

• Adding a custom field validation function

• Creating a basic Dublin Core form for collections

• Using Drupal taxonomies as controlled vocabularies
Adding a Custom Field Validation Function

1. Associate the function with the field

2. Add the function to a Drupal module
1. Associate the function with the field
2. Add the function to a Drupal module

function markslaptop_test_collection_genre_validate($element, &$form_state, $form) {

    if ($element['#value'] != 'Test genre') {
        form_error($element, t('You need to use "Test genre".'));
    }

}
Creating a Dublin Core Form for Collections

1. Create a new form
2. Create a root element
3. Create a title field
4. Create a date field
1. Create a New Form
2. Configure the Root Element
3. Create a dc:title Element
4. Configure dc:title
5. Create and Configure a dc:date Element
Preview Form

Title

This is my title.

This is the title of the object.

Date

2013-11-21

Enter the date that the object was created in the format YYYY-MM-DD.

Submit
Using Drupal Taxonomies as Controlled Vocabularies

1. Create a taxonomy in Drupal

2. Create a content type to attach the taxonomies to

3. In Form Builder, configure a textfield form element (e.g., MODS subject topic) to use the taxonomy as an autocomplete field
1. Create a Taxonomy in Drupal

Taxonomy is for categorizing content. Terms are grouped into vocabularies. For example, a vocabulary called "Fruit" would contain the terms "Apple" and "Banana".

Add vocabulary

<table>
<thead>
<tr>
<th>Vocabulary Name</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>languageTerm_code</td>
<td>edit vocabulary</td>
</tr>
<tr>
<td>languageTerm_text</td>
<td>list terms</td>
</tr>
<tr>
<td>marcgt</td>
<td>add terms</td>
</tr>
<tr>
<td>marc_country</td>
<td>edit vocabulary</td>
</tr>
<tr>
<td>marc_relations</td>
<td>list terms</td>
</tr>
<tr>
<td>_tags</td>
<td>add terms</td>
</tr>
<tr>
<td>typeOfResource</td>
<td>edit vocabulary</td>
</tr>
<tr>
<td></td>
<td>list terms</td>
</tr>
<tr>
<td></td>
<td>add terms</td>
</tr>
</tbody>
</table>
2. Populate the Taxonomy

You can reorganize the terms in TypeOfResource using their drag-and-drop handles, and group terms under a parent term by sliding them under and to the right of the parent.

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>cartographic</td>
<td>edit</td>
</tr>
<tr>
<td>mixed material</td>
<td>edit</td>
</tr>
<tr>
<td>moving image</td>
<td>edit</td>
</tr>
<tr>
<td>notated music</td>
<td>edit</td>
</tr>
<tr>
<td>software, multimedia</td>
<td>edit</td>
</tr>
<tr>
<td>sound recording</td>
<td>edit</td>
</tr>
<tr>
<td>sound recording-musical</td>
<td>edit</td>
</tr>
<tr>
<td>sound recording-nonmusical</td>
<td>edit</td>
</tr>
<tr>
<td>still image</td>
<td>edit</td>
</tr>
<tr>
<td>text</td>
<td>edit</td>
</tr>
<tr>
<td>three dimensional object</td>
<td>edit</td>
</tr>
</tbody>
</table>

Save Reset to alphabetical
3. Create a Content Type for Taxonomies

<table>
<thead>
<tr>
<th>LABEL</th>
<th>MACHINE NAME</th>
<th>FIELD TYPE</th>
<th>WIDGET</th>
<th>OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>title</td>
<td>Node module element</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Body</td>
<td>body</td>
<td>Long text and summary</td>
<td>Text area with a summary</td>
<td>edit delete</td>
</tr>
<tr>
<td>marcgt</td>
<td>field_marcgt</td>
<td>Term reference</td>
<td>Autocomplete term widget (tagging)</td>
<td>edit delete</td>
</tr>
<tr>
<td>lang_term_text</td>
<td>field_lang_term</td>
<td>Term reference</td>
<td>Autocomplete term widget (tagging)</td>
<td>edit delete</td>
</tr>
<tr>
<td>lang_term_code</td>
<td>field_lang_term_code</td>
<td>Term reference</td>
<td>Autocomplete term widget (tagging)</td>
<td>edit delete</td>
</tr>
<tr>
<td>typeofresource</td>
<td>field_typeofresource</td>
<td>Term reference</td>
<td>Autocomplete term widget (tagging)</td>
<td>edit delete</td>
</tr>
</tbody>
</table>

**Add new field**
- Label
- Type of data to store
- Form element to edit the data

**Add existing field**
- Label
- Field to share
- Form element to edit the data
4. Configure the Form Field

This is the machine name of the taxonomy field you added to your form.
Resources

• "How to Edit/Create Ingest Forms" Islandora Wiki. https://wiki.duraspace.org/pages/viewpage.action?pageId=45547834


sandbox.discoverygarden.ca

mleggott@discoverygarden.ca,
erin@discoverygarden.ca,
pround@upei.ca, &
mjordan@sfu.ca