REST Based Quality Control Reports

Tutorial

DSpace REST Report Tool Tutorial

The following repository contains a tutorial demonstrating the usage of the REST Base Report Tools: https://github.com/terrywbrady/restReportTutorial/blob/master/README.md

Summary

These reports utilize the DSpace REST API to provide a Collection Manager with

- an overview of their collections
- a tool to query metadata for consistency

When deploying the DSpace REST API, and institution may choose to make the API publicly accessible or to restrict access to the API.

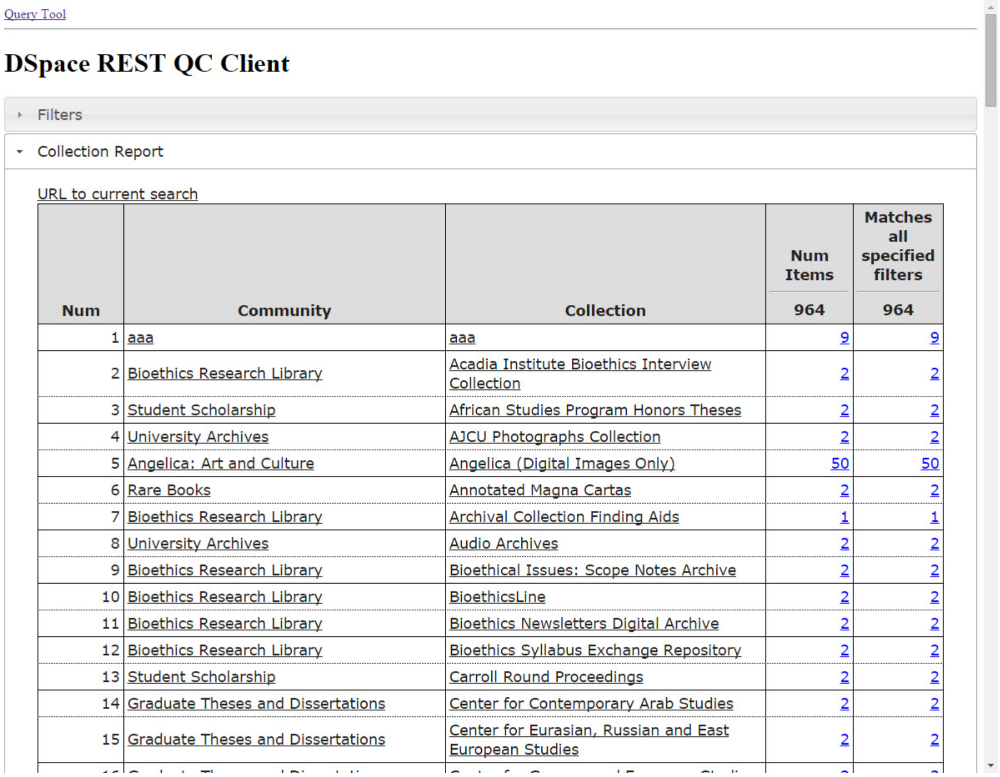
If these reports are deployed in a protected manner, the reporting tools can be configured to bypass DSpace authorization when reporting on collections and items.

API Calls Used in these Reports

REST Reports - Summary of API Calls

Report Screen Shots

Collection QC Report



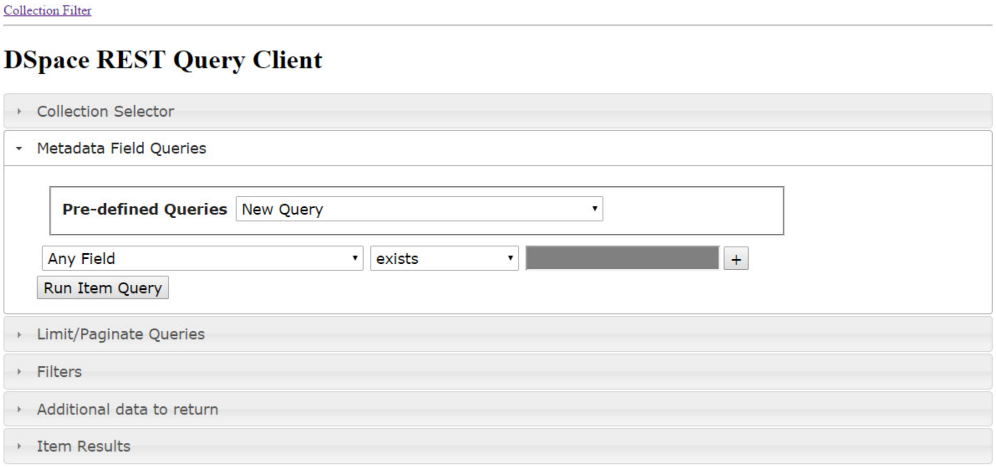
DSpace REST QC Client

URL to current search

Num	Community	Collection	Num Items 964	Matches all specified filters 964
1	aaa	aaa	9	9
2	Bioethics Research Library	Acadia Institute Bioethics Interview Collection	2	2
3	Student Scholarship	African Studies Program Honors Theses	2	2
4	University Archives	AJCU Photographs Collection	2	2
5	Angelica: Art and Culture	Angelica (Digital Images Only)	50	50
6	Rare Books	Annotated Magna Cartas	2	2
7	Bioethics Research Library	Archival Collection Finding Aids	1	1
8	University Archives	Audio Archives	2	2
9	Bioethics Research Library	Bioethical Issues: Scope Notes Archive	2	2
10	Bioethics Research Library	BioethicsLine	2	2
11	Bioethics Research Library	Bioethics Newsletters Digital Archive	2	2
12	Bioethics Research Library	Bioethics Syllabus Exchange Repository	2	2
13	Student Scholarship	Carroll Round Proceedings	2	2
14	Graduate Theses and Dissertations	Center for Contemporary Arab Studies	2	2
15	Graduate Theses and Dissertations	Center for Eurasian, Russian and East European Studies	2	2

REST Reports - Collection Report Screenshots with Annotated API Calls

Metadata Query Report



DSpace REST Query Client

Pre-defined Queries: New Query

Any Field | exists

REST Reports - Metadata Query Screenshots with Annotated API Calls

Installation and Configuration

Installing in DSpace 6

This code is part of the DSpace 6 code base.

Disabling the REST Reports

The REST reports will be enabled by default in DSpace 6. To disable the execution of these reports, remove the following line from dspace-rest/src/main/webapp/WEB-INF/web.xml

Enable/disable report resources in the REST API

```
<servlet-mapping>
        <servlet-name>default</servlet-name>
        <url-pattern>/static/*</url-pattern>
    </servlet-mapping>
```

Configuring Access of the Reporting Tools

The reports can be configured with anonymous access or the reports can be configured to bypass authorization checks.

Bypassing authorization checks allows collection owners to view the status of all items in the repository without authenticating through the REST API. This option is recommended **if you have secured access to your REST API**.

If your REST API is publicly accessible, deploy the reports with anonymous access and consider providing an authorization token for access to the report calls.

Configure Authorization for REST Reports

```
# Enable/disable authorization for the reporting tools.
# By default, the DSpace REST API will only return communities/collections/items that are accessible to a particular user.
# If the REST API has been deployed in a protected manner, the reporting tools can be configured to bypass authorization checks.
# This will allow all items/collections/communities to be returned to the report user.
# Set the rest-reporting-authenticate option to false to bypass authorization
rest.reporting-authenticate = false
```

Configure the REST Reports that can be requested by name

```
# Configure the report pages that can be requested by name
# Create a map of named reports that are available to a report tool user
# Each map entry should be prefixed with rest-report-url
#   The map key is a name for a report
#   The map value is a URL to a report page
# A list of available reports will be available with the call /rest/reports.
# If a request is sent to /rest/reports/[report key], the request will be re-directed to the specified URL
#
# This project currently contains 2 sample reports.  Eventually, additional reports could be introduced through this mechanism.
rest.report-url.collections = /rest/static/index.html
rest.report-url.item-query = /rest/static/query.html
```

Configure Item handle resolution

Enable the appropriate path to use to resolve an item handle **restReport.js**. (Depends on https://github.com/DSpace/DSpace/pull/1366/files)

Item Handle Resolution

```
    this.ROOTPATH = "/xmlui/handle/"
    //this.ROOTPATH = "/jspui/handle/"
    //this.ROOTPATH = "/handle/"
```

Enable User Authentication (Password AuthN only) for REST reports

Override the following function in your report file to enable/disable password AuthN for the REST reports. (Depends on https://github.com/DSpace/DSpace/pull/1369)

This setting can be found in **restReport.js**

Enable/Disable Password AuthN

```
        //disable this setting if Password Authentication is not supported
        this.makeAuthLink = function(){return true;};
```

Configure the database-specific format for a regex expression

```
# The REST Report Tools may pass a regular expression test to the database.
# The following configuration setting will construct a SQL regular expression test appropriate to your database
engine
rest.regex-clause = text_value ~ ?
```

Configure the sets of filters of interest to your repository managers

```
# A filter contains a set of tests that will be applied to an item to determine its inclusion in a particular
report.
# Private items and withdrawn items are frequently excluded from DSpace reports.
# Additional filters can be configured to examine other item properties.
# For instance, items containing an image bitstream often have different requirements from a item containing a
PDF.
# The DSpace REST reports come with a variety of filters that examine item properties, item bitstream
properties,
# and item authorization policies.  The existing filters can be used as an example to construct institution
specific filters
# that will test conformity to a set of institutional policies.
# plugin.sequence.org.dspace.rest.filter points to a list of classes that contain available filters.
# Each class must implement the ItemFilterList interface.
#   ItemFilterDefs:     Filters that examine simple item and bitstream type properties
#   ItemFilterDefsMisc: Filters that examine bitstream mime types and dependencies between bitstreams
#   ItemFilterDefsMeta: Filters that examine metadata properties
#   ItemFilterDefsPerm: Filters that examine item and bitstream authorization policies
plugin.sequence.org.dspace.rest.filter.ItemFilterList = \
        org.dspace.rest.filter.ItemFilterDefs,\
        org.dspace.rest.filter.ItemFilterDefsMisc,\
        org.dspace.rest.filter.ItemFilterDefsPerm

#     org.dspace.rest.filter.ItemFilterDefsMeta,\
```

Other filter configuration settings

The configuration file contains other settings that will control the behavior of the filters that you have enabled.

Enabling Sort-able Report Tables

1. Install sortable.js http://www.kryogenix.org/code/browser/sorttable/
2. Add to /dspace/modules/rest/src/main/webapp/static/reports
3. Include sortable.js in index.html and query.html

   Uncomment the following in index.html and query.html

   ```
   <!-- <script src="sorttable.js"></script> -->
   ```

4. Enable sortable in the report code in restCollReport.js and restQueryReport.js

CHANGE

```
var CollReport = function() {
    Report.call(this);
    //If sortable.js is included, uncomment the following
    //this.hasSorttable = function(){return true;}

var QueryReport = function() {
    Report.call(this);
    //If sortable.js is included, uncomment the following
    //this.hasSorttable = function(){return true;}
```

CHANGE TO

```
var CollReport = function() {
    Report.call(this);
    //If sortable.js is included, uncomment the following
    this.hasSorttable = function(){return true;}

var QueryReport = function() {
    Report.call(this);
    //If sortable.js is included, uncomment the following
    this.hasSorttable = function(){return true;}
```

Troubleshooting

Normalizing Metadata Language Field for Bulk Edit

The report tools allow a user to export query results as a CSV file that can be input with the Batch Metadata Editing tool.

If your repository has *unintentionally* tagged metadata fields with language codes, you may need want to normalize the language values in use.

Non-normalized View

Inconsistent use of language code

```
id,collection,subject[en],subject[en_us],subject[en_US]
111-222-333,123456789/1,cat,,
111-222-333,123456789/2,,dog,
111-222-333,123456789/3,,,bird
```

Normalized View

Consistent use of language code

```
id,collection,subject[en]
111-222-333,123456789/1,cat
111-222-333,123456789/2,dog
111-222-333,123456789/3,bird
```

The report tools contain a function that can be overridden to explicitly set a default language. Use this with caution in a multi lingual DSpace deployment.

Set language code on export

```
    this.getLangSuffix = function(){
      return "[en]";
    }
```

The following SQL might be helpful in this instance.

Sample SQL to Normalize Language - Use with Caution

```
update metadatavalue set text_lang='en'
where text_lang is null and dspace_object_id in (select uuid from item);

update metadatavalue set text_lang='en'
where text_lang in ('','en_US', 'en-US','en_us')
and dspace_object_id in (select uuid from item);
```

Installing in DSpace 5

This feature is not a part of the DSpace 5 code base. Please see the following notes to enable a DSpace 5 compatible version of these reports.

1. Install https://github.com/DSpace/DSpace/pull/1568
2. Change the following code into restCollReport.js and restQuery.js to pull the correct id for each DSpace Object

   Change the following in restCollReport.js and restQuery.js

   ```
   var CollReport = function() {
   Report.call(this);
   var QueryReport = function() {
   Report.call(this);
   ```

   Change TO

   ```
   var CollReport = function() {
   Report.call(this);
   this.getId = function(obj) {return obj.id;}
   var QueryReport = function() {
   Report.call(this);
   this.getId = function(obj) {return obj.id;}
   ```