Troubleshooting a Batch Ingest

Introduction

Table of Contents:

A Batch Ingest process may fail for a variety of reasons, after which a user may replay the Manifest File with corrected information. This guide explains the different reasons why a spreadsheet may fail, how the errors can be corrected, and how to replay a Manifest File.

To learn how to create a basic Manifest File and start a Batch Ingest, see Uploading Content using Batch Ingest

Definitions

- **Batch Ingest:** the process of creating multiple media items by uploading an Ingest Package to an Avalon dropbox
- **Ingest Package:** the combination of content and metadata, consisting of one Manifest File and one or more Content Files
- **Manifest File:** a spreadsheet containing the metadata for the items to be created as well as the names of the Content Files that make up each item
- **Content File:** a single media file that is part of an item; one or more Content Files can make up an item in the Avalon Media System
- **Item:** a single media object accessible through one Avalon page; consists of one or more media files and metadata describing the media file (s)
- **Collection:** a grouping of items in Avalon for administrative and discovery purposes. Items belong to one and only one collection

Preliminaries

When a new collection is created, Avalon creates a sub-directory with the name of that collection (substituting underscores for any blanks) within the Avalon dropbox directory. A Batch Ingest is initiated by uploading an Ingest Package to the collection sub-directory in the dropbox. To connect to the Avalon dropbox, see Uploading Content to an Avalon Dropbox.

The Manifest File is a spreadsheet (xls, xlsx, csv, or ods) containing the metadata for the items to be created, as well as the file paths to the content files that make up each item; download manifest_example.xlsx for a blank manifest template. The Manifest File lists many metadata fields, but only a few of these fields are required: "Title," "Date Issued," and "File." However, "Bibliographic ID" may be used in place of "Title" and "Date Issues." For a description of the other fields, as well instructions for adding structure or captions, see Batch Ingest Package Format.

Troubleshooting a Failed Ingest

Invalid Manifest File

Avalon will check for new spreadsheets once every minute. Upon detection, the system will open the spreadsheet and attempt to validate it for correct information. Spreadsheets may fail validation for the following reasons:

- File is corrupt
- No username listed
- The listed username account does not have permission to upload to the collection

If the user account exists in the Avalon system, but the spreadsheet is invalid for some reason, the user will be emailed a failure message with explaining why the spreadsheet was not accepted; multiple reasons may be listed.

Example error message:

```
User USER_KEY does not have permission to add items to collection: COLLECTION_NAME
```

If the user account does not exist in the Avalon system (i.e. username was left blank, entered incorrectly, or no email address is associated with the account), the failure message will be sent to the notification email address for Avalon. The default notification email address is "avalon-notification@example.edu," and can be located in `Settings.yml`:

```
email:
  comments: 'avalon-comments@example.edu'
  notifications: 'avalon-notifications@example.edu'
  support: 'avalon-support@example.edu'
```

Avalon on AWS

If your instance of Avalon is running on AWS, default emails will need to be verified with AWS first (by clicking a verify link in a message sent to the email address).

Valid Manifest File

If the spreadsheet passes validation, the user will receive an email stating that their spreadsheet has been accepted:

```
Your metadata package was validated and no errors were found. Your batch is now queued.
Manifest file: example_manifest_file.xlsx
```

This message does not mean that the ingest was successfully completed, only that the sheet was accepted and the rows have been queued for ingest.

Every fifteen minutes, a task will run to check if a Batch Ingest has finished (all rows have been run and were either ingested or failed due to an error). Upon detecting a finished Batch Ingest with any errors, the user will receive email with a report containing a replay name in the format *universally-unique-id_original_filename*.

Example replay name:

```
2a29d341-7da8-44c1-a503-a0df44fd0337_example_manifest_file.xlsx
```

Example batch ingest report, with reported errors:

Initial processing of your batch ingest failed. Details are below. Please see the Batch Ingest Package Format for further help.

To replay this batch, please rename your spreadsheet to e2cc9d52-335f-4f3f-bd4f-409851adf218_no_title.xlsx and reupload it after making desired changes. Only unpublished items can be updated via this method. Retain all rows in the spreadsheet including those that have no changes.

The following rows of your spreadsheet failed:

Row	Error
3	To successfully ingest, either title and date issued must be set or a bibliographic id must be provided

Any reported errors will always begin at Row 3 (Row 1 contains username information, Row 2 contains headings). Use the row number to find the object causing an error, and use the error description to correct the object information. Refer to Batch Ingest Package Format for the required syntax for each field.

Every twenty-four hours, a task will check for Batch Ingest processes with no activity since the last check. Activity includes changes to the batch registry (username, collection, etc.) and changes to individual entries (entries succeeding or failing). For processes that have been inactive for 4 days or longer, an alert email will be sent to the notification email address for Avalon for all batches with no activity. This alert is informational only; it does not imply that anything is wrong.

Replaying a Manifest File

By using the replay name contained in the report email, Avalon will rerun a corrected spreadsheet and ingest any previously failed items. Replaying will also update any ingested and unpublished media objects that have received changes. **Please note that published media objects may not be corrected using a replayed Manifest File, so this method will always fail if Avalon is auto-publishing media objects.**

To replay a Manifest File:

1. Rename the original Manifest File to the replay name provided in the report email
2. Change any data that needs to be updated
    a. Provide missing or incorrect data that caused an error
    b. Update data for successfully ingested media objects (i.e. if the data didn't cause an error, but needs to be edited)
3. DO NOT delete rows that will remain unchanged; leave them in the Manifest File as is
4. Upload the Manifest File using the normal process
5. Re-upload the Content Files if they were removed for any reason (e.g. by automatic processes)

After Avalon detects the uploaded Manifest File, it will queue the rows for ingest as it normally does. When it reaches a changed row, it will take the following actions:

- For previously failed rows, the row will rerun for ingest

- For previously ingested items, it will check if the media object is currently published:
    - For published items, the row will be marked as in error (published items may not be corrected using a replayed Manifest File)
    - For unpublished items, the ingested media object will be deleted, and a new media object will be created using the data from the row

If necessary, replayed Manifest Files can be stacked, although it isn't advisable to do so. After a Manifest File has been corrected and re-uploaded, a user may make another correction and re-upload immediately again. Doing so will cause a changed media object to be processed twice (once for each replayed spreadsheet).

Configuring Batch Ingest

The data for Batch Ingest is stored within an SQL database. Manifest Files are registered in the BatchRegistries table, and individual rows are stored in the BatchEntries table, linked by a key. Admins can view these tables to understand how the spreadsheets are ingested and how information is stored. See the `db/schema.rb` file for more information on these tables.

The different tasks that schedule Batch Ingest can be found in the `config/schedule.rb` file, and their times can be altered as desired:

- `avalon:batch:ingest` - checks for new spreadsheets
- `avalon:batch:ingest_status_check` - checks for completed batches, sends report emails
- `avalon:batch:ingest_stalled_check` - checks for stalled batches (no activity in 4 days), sends report emails

After a new spreadsheet is detected and validated, an IngestBatchEntryJob is queued as `:ingest` (for those who want to throttle jobs in the queue). If the job queue is lost, every row in BatchEntries with a `current_status` of `Queued` will need to be re-added to the job queue. The job is documented and defined in `app/jobs/ingest_batch_entry_job.rb`.