EAD-XML Finding Aid Inventory Creation Scripts Usage Guide

Purpose

This guide documents the creation of encoding new finding aids using the EAD XML Conversion Script. It also documents the revision of existing encoded finding aids that do not contain normalized ISO8601 date values using the EAD XML Legacy Conversion script at the SWC/SCL.

Table of Contents

Download and Installation Instructions, p. 2
New Finding Aids, p. 3
Legacy Finding Aids, p. 9

Rights Statement

Download and Installation Instructions

GitHub

Click the green "Clone or download" button and download the GitHub page as a zip/rar/other file.

Email:

If a computer blocks downloading the script from email

- The script likely appears as a .txt file or something similar. Click the text file and click select "Save As."
- Change the file format in the "Save As" window from "text file" to "all documents". Do not rename the script file yet.
- Save the script file somewhere it can be easily accessed, because it will save all generated xml code to the folder in which the script is housed.
- Once saved, the script should appear as a white page icon with a blue computer screen icon on top with a > symbol on the screen. This appearance means it converted to the proper format. At that time, if it is not already named the "EAD-XML Conversion Script" or "EAD-XML Legacy Conversion Script," name the file as appropriate.

Excel PowerShell Module Installation

Requires PowerShell version 5 and an internet connection.

The script uses PowerShell 5, which comes standard on a regularly-updated Windows 10 OS. Before running the script, however, you must install a PowerShell module before the code will run.

1. Click Windows Start Menu
2. Type PowerShell
3. Right Click Windows PowerShell
4. Click Run as Administrator
5. Type **Install-Module ImportExcel -Force**
6. Accept all prompts asking if you would like to proceed
7. The module should install

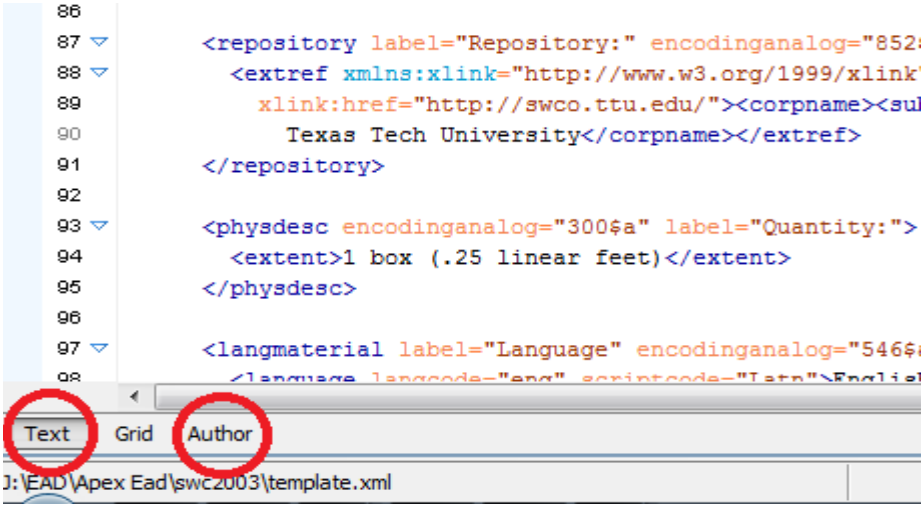
New EAD-XML Inventory Creation Guidelines

This section documents the SWC/SCL EAD Inventory Creation guidelines using the NewEADXMLCreationScript.

Other methods of inventory creation, such as line-by-line encoding or Microsoft Word's Mail Merge function, are not covered. They should no longer be used to create EADs at the SWC/SCL.

Programs to Use

- **MS Excel** – to create inventories of collections that will later be used as the data set to generate XML code for EAD inventories.
- **PowerShell 5** – to generate XML code by running a PowerShell script on the Excel document.
    - Version 5 is required. It comes standard on a properly updated Windows 10 as of 9/2018.
    - If this is a user's first time running the script, they will need to:
        - Search for PowerShell in the Windows Start Menu
        - Right click PowerShell and select "Run as Administrator"
        - Type the following text: **Install-Module ImportExcel -Force**
        - Accept all prompts asking the user to proceed
- **Oxygen XML Author** – to interact with XML finding aid documents.
    - The Text view in Oxygen is suitable for adding code, inputting metadata, and ensuring the XML is correctly formed
    - The Author view is useful for finding spelling and grammatical errors in inventories and collection level metadata, as well as inconsistences in inventories such as missing dates or other information.



File Naming Structure

- As a best practice, use consistent file names in your Excel inventories.
    - Names should include the name of the Papers/Records/Collection document in the inventory, as well as the nature of the inventory (linked or non-linked)
        - Identify whether or not the inventory will use hyperlinks in its XML code (i.e. which Word template to use).
        - Example:
            - Use the name **R.G.Carter_inventory** for a non-linked inventory
            - Use **R.G.Carter_linked_inventory** for an inventory that contains links to a corresponding DSpace collection
- Templates are named:
    - Excel: **(department) XML Database template – DateCompliant**

Inventory Creation Instructions

DO NOT CHANGE COLUMN HEADERS in the Excel inventory template. Column headers are triggers for the script to generate valid XML code.

Type the inventory for the collection in the Excel template provided

- In the "Attribute" column, input "series" for series, "subseries" for sub-series, and "file" or "item" for objects at the folder or item level
    - **All** c01-level entries are "series"
    - "subseries" will only occur in c02, c03, etc. entries that are not a folder-level entry
- In the "c0#" column, input the numerical value that corresponds to the component level of the XML code
    - In English: input a 1 for <c01>, 2 for <c02>, and so forth
- In the "Box" column, type the box number of the object described in the "Title" field
    - Series and sub-series will have a blank "Box" value
- In the "File" column, type the folder or item number of the object described in the "Title" field
    - Series and sub-series will always have a blank "File" value
- In the "Title" column, type the name of the file folder, artifact, framed item, or other media that corresponds to the "Box" and "File" values
    - Series and sub-series values will also be entered here
    - **Type commas after every folder title**
    - **Do not** type commas after Series or Sub-series
- In the "Date" column, type the folder or item's date **as the patron will read it**.
    - Per **DACS 2.4**, dates should not be abbreviated:
        - **Valid:** January 1956
        - **Invalid:** Jan. 1956
        - Values of "undated" should be entered into the "Date" column. Failing to do so will result in no date value displayed fin the finding aid.

        - When the Creation Script runs, it will account for a wide number of permutations for dates, but can fail if entries are too unusual. Therefore, at all times use the **Valid** date format above:
        - **The Conversion Script cannot account for Non-alphanumeric characters, such as quotation marks, question marks, parentheses, forward and back slashes, colons, brackets, and a host of others. Avoid these whenever possible.**
        - For values of "undated", the Conversion script generates a value of the entire date range of a collection, based on the highest and lowest values in the finding aid, just as we do manually when we determine the date range of the collection.
            - For example: the "John Perrin Papers, 1935-2010 and undated" would have the Conversion Script generate a value of "1935/2010", i.e. the known date range for the collection
    - Date fields in Excel should always be set as "Text" and not as "Date" or "Number" in the "Format Cells > Number" menu in Excel.
        - Date fields are set as "Text" by default. If errors or automated reformatting occur, that may be the result of inadvertently changing the field back to "Date"
    - **If you do not follow the guidelines above, or other formats as stated in DACS 2.4, then the script WILL cause errors in Oxygen, or worse, incorrect @normal values.**
- In the "DateCoded" column, **it is no longer necessary to input values**. The Conversion Script will automatically create normalized date values in the XML code for every instance wherein the "Date" column contains a value.
- In the "DSpace URL" column, input the URI or "handle" of the folder or item on DSpace to which the folder or item in the Excel inventory corresponds.
    - The Conversion Script will identify inventory entries that require <extref ></extref> (i.e. the code that creates hyperlinks when the finding aid is displayed on a website.) It will generate the appropriate XML code for that folder, but will not do so for non-linked folders.
    - In other words, the Conversion Script can create code for finding aids that contain both linked and un-linked folders
- The Conversion Script now ensures ampersands appear in the XML code by converting any "**&**" characters in the Excel document to **&**
- **Be careful to save the inventory file under the inventory name you wish to use, rather than mistakenly saving over the template!**

Running the Script

1. Once the Excel inventory is complete, right click on the script entitled **NewEADXMLCreationScript**

2. Select "Run with PowerShell"

3. The PowerShell window will open, as seen below.

4. A Windows navigation box will also open. Inside it, navigate to the Excel inventory spreadsheet that you want to convert. Select that inventory, and click Open.

5. The PowerShell program will begin to run the Conversion Script. You may see a loading/progress bar if the collection inventory is very large.

6. A Notepad document entitled "xmlOutput" followed by a string of numerals will open. The document should display the entirety of your XML code.
    - An XML file (*not* a Notepad file) containing the same code has automatically been saved to the folder in which the Script is located, **which is not necessarily the folder in which the Excel inventory is located.**
    - The XML file shares the name of the Notepad document.
    - In some rare cases, once the conversion is complete the message "Processing Complete. Press enter to close." will appear in the PowerShell window. Press Enter, and the box will close. Afterward, the Notepad document will appear.

7. In the Notepad document (or if you have inadvertently closed it, in the XML file of the same name), "Select All" (or Ctrl+A) of the text and Copy it.

8. Open the collection's XML file in Oxygen, navigate to the **<dsc>** element, or "Collection Inventory," and paste the inventory XML code.

9. Quality Assurance process
   - If Series ID (id="ser#") was entered into the spreadsheet, then it will be present in the Excel inventory template in every **<c01>** value
     - **If duplicate values were entered (i.e. "1" appears more than once in the Series ID column), then Oxygen will generate an error.**
       - Correct that manually in Oxygen
       - Or in the Excel inventory, which will require re-running the Creation Script and re-pasting its contents into Oxygen.
   - If any critical columns were left blank in the Excel inventory, the Creation Script will notify the user that an error occurred, will terminate the script before Creation is complete, and delete the Notepad/XML document it was creating so that it does not get used.
     - In these instances, correct the errors in the Excel inventory, then run the script again.
   - Although the Conversion Script can convert a large number of permutations of "Date" values into normal="" values, discrepancies can occur. A thorough spot check of normal="" values, particularly originating from "undated" values in the <unitdate > element, is strongly advised. **NOTE: if a user is following 'Date' data entry formatting per DACS 2.4, the majority of the errors below will not occur. Errors may include:**
     - Non-alphanumeric characters, such as quotation marks, question marks, parentheses, forward and back slashes, colons, brackets, and a host of others will almost certainly result in errors. Errors should, but not always are, visible in Oxygen XML Author:
     - Incorrect @normal values for any dates containing the above characters.
     - Incorrect @normal values for "undated" date values, or any others that should contain the complete date range of the collection (see "What the…Does", above)
     - A value of "nd" rather than "undated" will produce an error
       - Follow DACS 2.4
     - Blank <unitdate> fields will produce an error.
       - This is because if there is no date attached to the component
     - Values of "day and day month year" (eg. 12 and 20 March 1976) will display errors in @normal values in Oxygen. Similar irregular permutations will do the same.
       - NOTE: in some instances no error will appear, but the @normal value will be erroneous, but still validated by Schema or other validation files. (eg. "11 September 1992" became "normal=1992"
       - Follow DACS 2.4
     - Values of "year to year" (eg. 1916 to 1969) will produce erroneous @normal values
   - Reviewing just one instance of the @normal value for an "undated" folder/item/box to verify that it included the correct Inclusive Dates is sufficient to confirm that it is correct throughout the document.

- The Conversion Script cannot account for italics or bold text. Instances where that is required will need to be coded manually. The XML code for italics and bold are, respectively:

  ```
  <emph render="italic">TEXT</emph >
  <emph render="bold">TEXT</emph >
  ```

- **Pro Tip:** Using the "**Author**" tab at the bottom of the screen in Oxygen makes it far easier to find empty Date, Box, and Folder tags that need to be deleted from the code

Legacy EAD-XML Creation Guidelines

NOTE: When using the LegacyEAD-XMLConversionScript, converted XML files will appear in a folder titled "edit" that is automatically created in the folder wherein the Legacy Script was run.

What the LegacyEAD-XMLConversionScript Does and Doesn't Do

- Adds @normal values to everything that has <unitdate>, even components that have multiple <unitdate> tags
    - Items listed solely as "undated" (per DACS 2.4) will receive an @normal value inclusive of the earliest and latest dates found anywhere in the <dsc> (inventory) section of the XML code.
- Deletes <frontmatter> from collection level inventory of all finding aids
    - This complies with Texas Archival Resources Online (TARO) metadata best practices.
- DOES NOT add @level "series" to <c01> components as of this version, nor any other @level values throughout the finding aid
- DOES NOT add @id ser# to components as of this version

Errors Should Not Occur When:

- A @id value (ser#) is already present in <c01>
- A date contains a superscript (eg. October 1st, 2nd, etc.)
- Commas exist after data in <unittitle>, or before data in <unitdate>
- @normal is already present
- Either "month year" (per DACS 2.4) or "month, year" (i.e. there's a comma where it should be) was used
- NOTE FOR EAD3: adding @normal will not solve prepare a finding aid for EAD3 conversion.
    - Because EAD3's new <unitdatestructured> requires one, and only one, of 3 elements within it, @normal will rarely reflect the folder/item/box dates

Common Errors

NOTE: Following 'Date' data entry formatting per DACS 2.4 will prevent the majority of the following errors.

- Non-alphanumeric characters, such as quotation marks, question marks, parentheses, forward and back slashes, colons, brackets, and a host of others within the <unitdate> tag will almost certainly result in errors. The script may complete conversion, but will likely have created the following errors, visible in Oxygen XML Author:
    - Incorrect @normal values for any dates containing the above characters
    - Incorrect @normal values for "undated" date values, or any others that should contain the inclusive date range of the whole collection
- "nd", "n.d.", or similar values other than "undated" will produce an error
    - Follow DACS 2.4
- Blank <unitdate> fields will produce an error.

    - This is because if there is no date assigned to a file/item/box, it should not have a <unitdate> tag in the first place.
- Values of "day and day month year" (eg. 12 and 20 March 1976) will display errors in @normal values in Oxygen. Similar irregular permutations will do the same.
    - HOWEVER: in some instances no error will appear, but the @normal value will be erroneous. It will still validate by schema or other validation rules.
        - Example: "11 September 1992" becomes "normal=1992"
    - Follow DACS 2.4
- Values of "year to year" (eg. 1916 to 1969) will produce erroneous @normal values containing the word "to"

Programs to Use

- **PowerShell 5** – to generate XML code by running a PowerShell script on the Excel document.
    - Version 5 is required. It comes standard on a properly updated Windows 10 as of 9/2018.
    - If this is a user's first time running the script, they will need to:
        - Search for PowerShell in the Windows Start Menu
        - Right click PowerShell and select "Run as Administrator"
        - Type the following text: **Install-Module ImportExcel -Force**
        - Accept all prompts asking the user to proceed
- **Oxygen XML Author** – to interact with XML finding aid documents.
    - The Text view in Oxygen is suitable for adding code, inputting metadata, and ensuring the XML is correctly formed
    - The Author view is useful for finding spelling and grammatical errors in inventories and collection level metadata, as well as inconsistences in inventories such as missing dates or other information.

EAD XML Conversion Script Instructions:

1. Once the Excel inventory is complete, right click on the script entitled **LegacyEADXMLConversionScript**

2. Select "Run with PowerShell"

3. The PowerShell window will open, as seen below.

4. Users are given the option to type "1" to process a single XML file or "2" to process a batch of XML files. Select the number appropriate to your task.

5. A Windows navigation box will then open. Inside it, navigate to the .xml file(s) that you want to convert. Select the file(s), and click Open.

6. The PowerShell program will begin to run the Conversion Script. You may see a loading/progress bar if the collection's inventory is very large.
    - Of a progress bar is not present, please be patient as the script is still running.

7. Users will receive **NO NOTIFICATION** that the process has been complete, but it has been completed when the message "Press Enter to continue" appears.

8. The converted XML file has been automatically saved to a folder entitled "edits" located in the same folder as the XML file that was updated. The title of the file will be the title of the original .xml file followed by a long string of numerals.
    - For example, if the original file was entitled "00001.xml", the output file would be entitled something similar to:

      00001.xml_NEW-8586371711816787952

9. Final Review of the EAD XML document
    - Are there any 'red marks' on the right side of the Oxygen Author "Author" screen even though?
        - This will occur if the appropriate Schema, CSS, stylesheet, or other validation files are not present in the same folder as the newly converted .xml file
    - Now examine the "Common Errors" section (page 9)
        - Red mark by red mark, reconcile the dates.