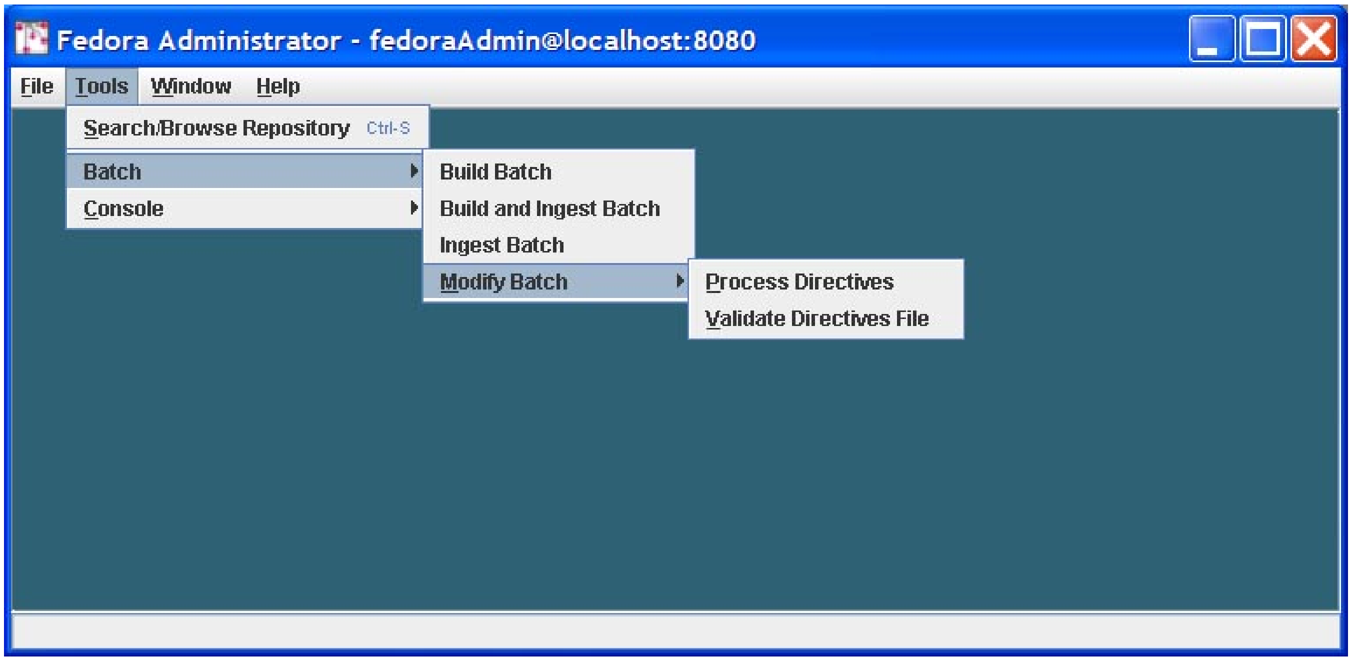
Batch Processing

Batch Processing

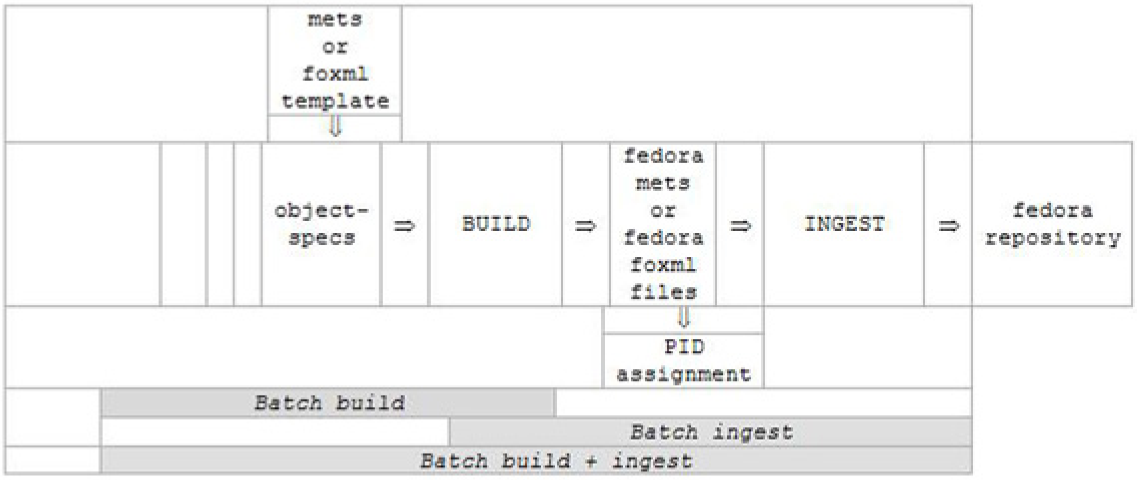


The Batch menu item includes tools to create and ingest multiple Fedora objects, which are FOXML documents or Fedora-specific METS documents contained in files outside the repository. The Batch Modify tool is also included here and allows users to modify batches of already-ingested Fedora objects.

Batch Ingest

It's simple to ingest objects created by one-up edit or by custom scripting. The Batch menu also supports building objects. This takes a general template common to all objects in a batch and makes object-specific substitutions into the template. The template can be either a Fedora METS XML document or a Fedora FOXML document and contains data common to the objects of the batch. Separate XML documents hold the per-object substitution values. The format of the template (FOXML or METS) determines the format of the built objects. The relatedness of objects in a batch is defined by what Fedora Administrator allows to be substituted and by which substitutions you choose to make. Data from the template are retained, unless replaced per individual object, including XML comments.

Fedora Administrator provides for three modes of object batch processing: **batch build**, **batch ingest**, and a combined **batch build and ingest**. This phased processing is shown in the following diagram:

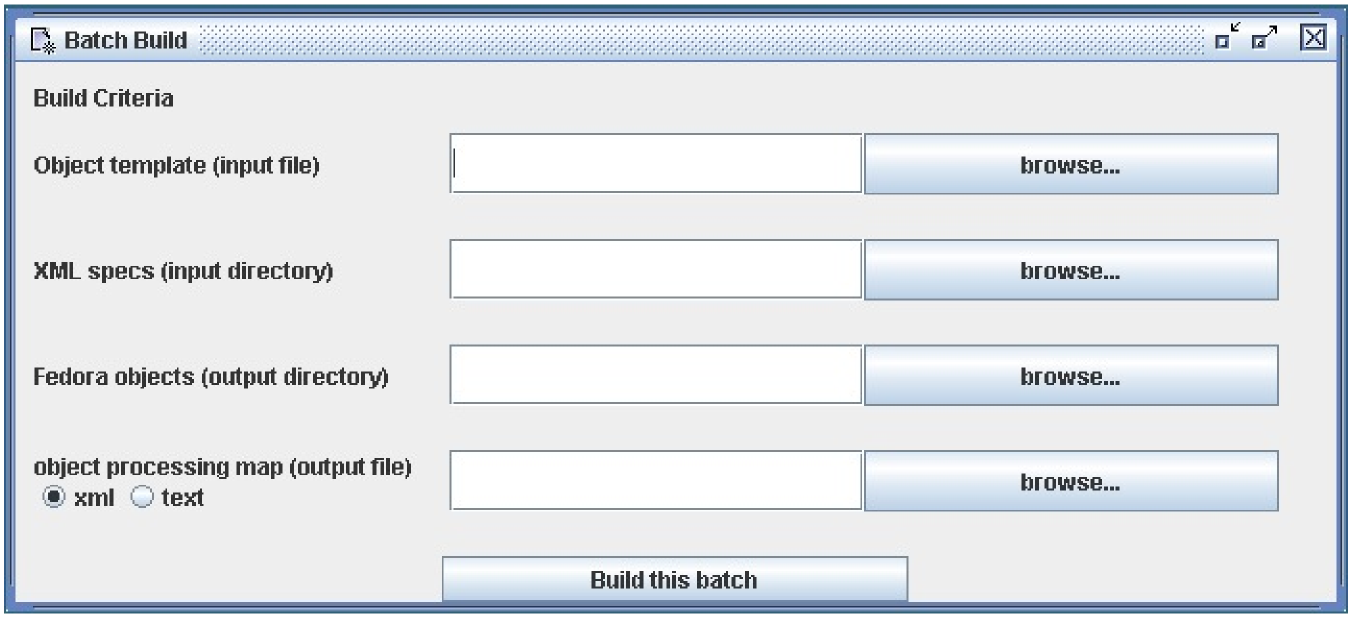


Building Fedora objects in Batch

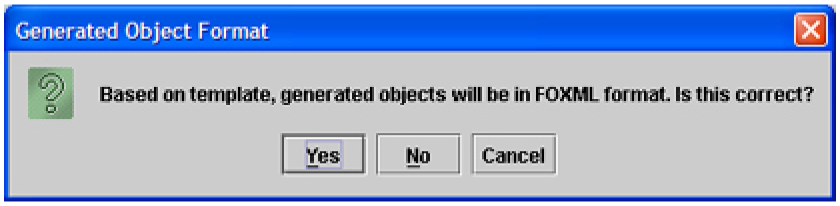
Build a set of Fedora METS XML or Fedora FOXML XML files from a common Fedora METS or Fedora FOXML template and simple (non-METS) XML *obj ect-specs*. The resulting objects are then ready for ingesting into Fedora.

Fedora Administrator Instructions

Select *Tools* on the Fedora Administrator menu bar, select item *Batch* and then *Build Batch._This will open a _Batch Build* window. You may need to adjust this window's size to see its controls. Use the *browse* buttons to enter the four required settings. Clicking on a *browse* button opens a standard directory/file selection dialog.
Then click the *Build this batch* button to build the batch of Fedora *METS* or Fedora *FOXML* XML documents.

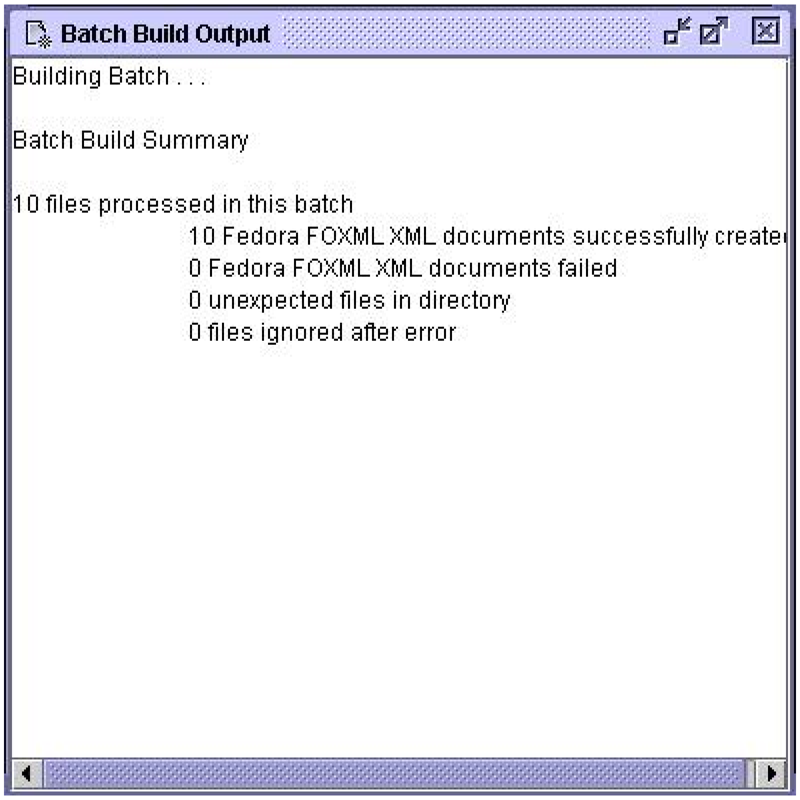

Batch Build
Build Criteria
Object template (input file)
browse...
XML specs (input directory)
browse...
Fedora objects (output directory)
browse...
object processing map (output file)
● xml ○ text
browse...
Build this batch

A confirmation dialog will open requesting confirmation of the object template selected. Clicking *Yes* will continue the batch build. Clicking *No* or *Cancel* will return the user to the *Batch Build* window.


Generated Object Format

Based on template, generated objects will be in FOXML format. Is this correct?

A second (output-only) window will open to show progress. You can build multiple different batches before closing the *Batch Build* window.



```
Building Batch . . .

Batch Build Summary

10 files processed in this batch
                10 Fedora FOXML XML documents successfully create
                0 Fedora FOXML XML documents failed
                0 unexpected files in directory
                0 files ignored after error
```

You can then ingest the created batch as described elsewhere in this document. No subdirectories or files are deleted by Fedora Administrator. Setup and cleanup of the files in the batch must be done by you using standard operating systems facilities.

To Demo

You can use files and subdirectories of directory **client/demo/batch-demo**, relative to your **FEDORA_HOME** environment variable. (When you create your own batches, the needed directories and files can be anywhere in the file space of the system on which you are running Fedora Administrator or command-line *BatchTool*.)

- Use file **mets-template.xml** for *METS template* (*input file*) if you want to build *METS* objects.
- Use file **foxml-template.xml** for *FOXML template* (*input file*) if you want to build *FOXML* objects.Use subdirectory **object-specifics** for *XML specs* (*input directory*); this is a directory holding (all and only) per-object data.
- Use subdirectory **objects** for the built objects (*output directory*); this is a directory to hold (all and only) Fedora object files built by Fedora Administrator. The format of the built files will be determined by the type of template used (*METS* or *FOXML*).

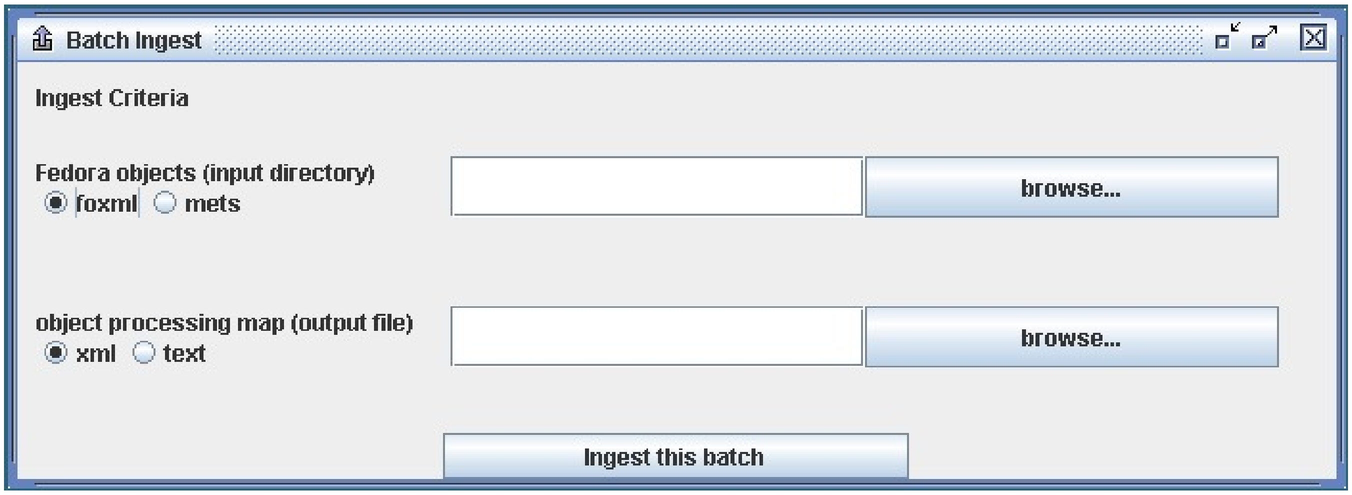
Specify a file path of your choice for *object processing map* (*output file*); this is a file which maps *object-specs* to objects built. See the section on *object processing maps*, elsewhere in this documentation. Note that PIDs cannot be reported in this (*Batch Build*) mode, as they have not yet been assigned. Optionally select the output format for *object processing map*, either **xml** or **text** (**xml** is the default format).

Ingesting Fedora Objects in Batch

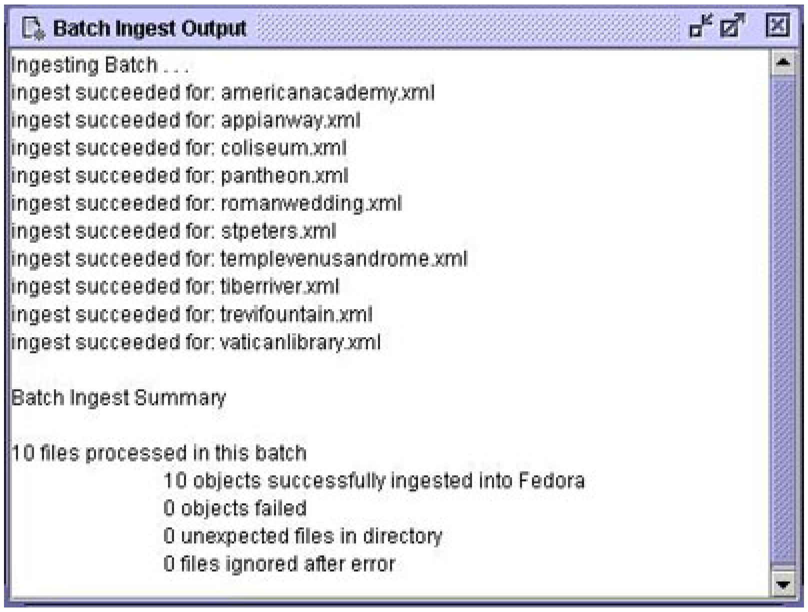
Create a set of Fedora objects in your repository from a corresponding set of Fedora *METS* XML or Fedora *FOXML* XML files.

Fedora Administrator Instructions

Select *Tools* on the Fedora Administrator menu bar, and select item *Ingest Batch*. This will open a *Batch Ingest* window. You may need to adjust this window's size to see its controls. Use the *browse* buttons to enter the two required settings. Clicking on a *browse* button opens a standard directory/file selection dialog.



You must also indicate the format of the objects to be ingested by selecting either the foxml or mets radio button. Then click the *Ingest this batch* button to ingest the batch into your Fedora repository. A second (output-only) window will open to show progress. You can ingest multiple different batches before closing the *Batch Ingest* window.



```
Ingesting Batch . . .
ingest succeeded for: americanacademy.xml
ingest succeeded for: appianway.xml
ingest succeeded for: coliseum.xml
ingest succeeded for: pantheon.xml
ingest succeeded for: romanwedding.xml
ingest succeeded for: stpeters.xml
ingest succeeded for: templevenusandrome.xml
ingest succeeded for: tiberriver.xml
ingest succeeded for: trevifountain.xml
ingest succeeded for: vaticanlibrary.xml

Batch Ingest Summary

10 files processed in this batch
                    10 objects successfully ingested into Fedora
                    0 objects failed
                    0 unexpected files in directory
                    0 files ignored after error
```

No subdirectories or files are deleted by Fedora Administrator. Setup and cleanup is by using standard operating systems facilities. Fedora Administrator does not itself validate on *Batch Build*, but batch ingest into Fedora does. The batch fails on the first individual object ingest failure. Fedora will not ingest a *METS* file whose **METS:xmldata** elements are empty or contain non-tagged character data.

To Demo

You can use files and subdirectories of directory **client/demo/batch-demo**, relative to your **FEDORA_HOME** environment variable. (When you create your own batches, the needed directories and files can be anywhere in the file space of the system on which you are running Fedora Administrator or command-line *BatchTool*.) You will need to have already done a *Build Batch* demo, explained elsewhere in this document, to populate the **objects** directory needed in this current demo. If you have ingested these objects before, either in this *Ingest Batch* mode following a separate *Build Batch* mode, or in a *Build and Ingest Batch* mode, you will first need to edit **OBJIDs** in the *object-spec* files, or to remove the corresponding objects from your Fedora repository.

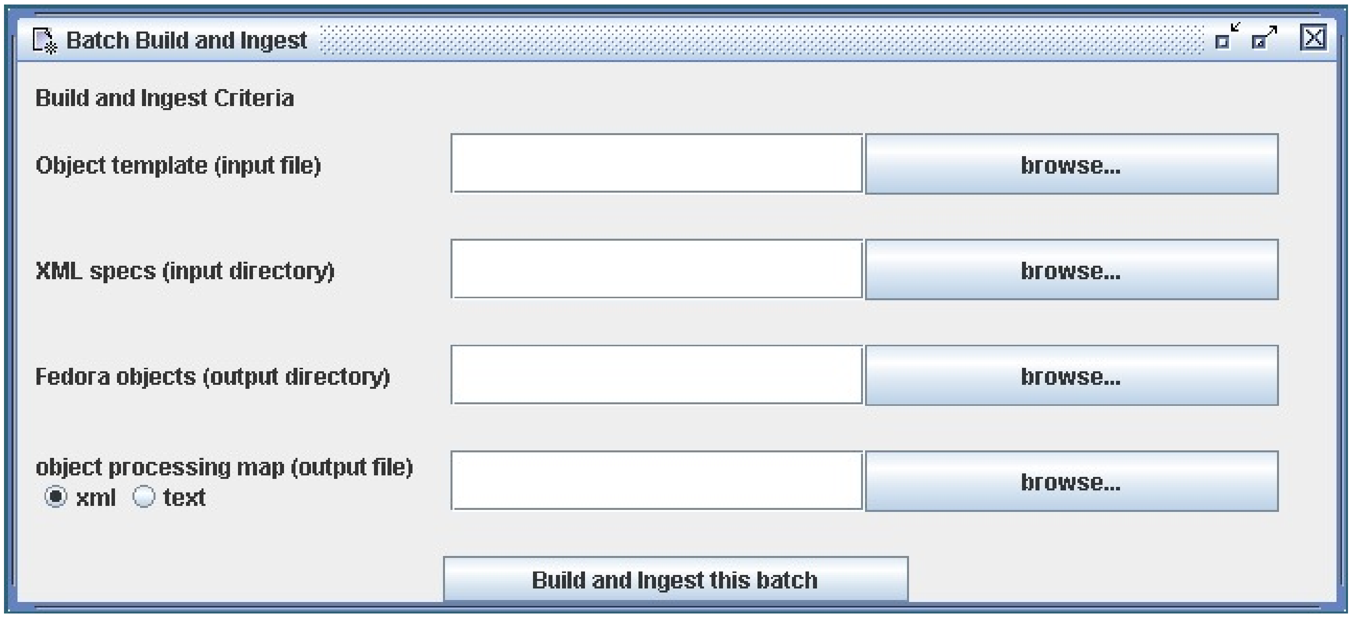
Use subdirectory **objects** for *built objects* (*input directory*); this is a directory holding (all and only) Fedora object files to ingest.Specify a file path of your choice for *object processing map* (*output file*); this is a file which maps objects to their assigned PIDs. See the section on *object processing maps*, elsewhere in this documentation. Note that *object-specs* of objects previously built by Fedora Administrator cannot be reported in this (*Batch Ingest*) mode, as they (as source documents) are no longer known. Optionally select the output format for object processing map, either **xml** or **text** (**xml** is the default format).

Building and Ingesting Fedora objects in Batch

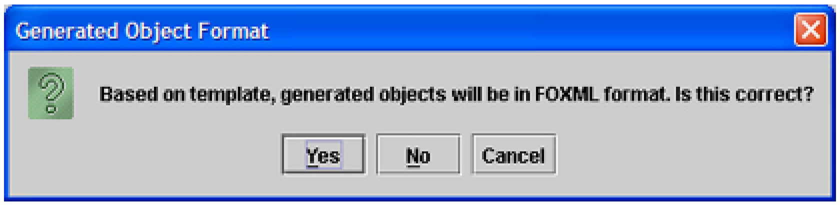
This process builds a set of Fedora *METS* XML or Fedora *FOXML* XML files from a common Fedora *METS* or Fedora *FOXML* template and simple *batchMerge* XML *object-specs*, then ingests the resulting batch into Fedora.

Fedora Administrator Instructions

Select *Tools* on the Fedora Administrator menu bar, and select item *Build and Ingest Batch*. This will open a *Batch Build* and Ingest window. You may need to adjust this window's size to see its controls. Use the *browse* buttons to enter the four required settings. Clicking on a *browse* button opens a standard directory/file selection dialog.

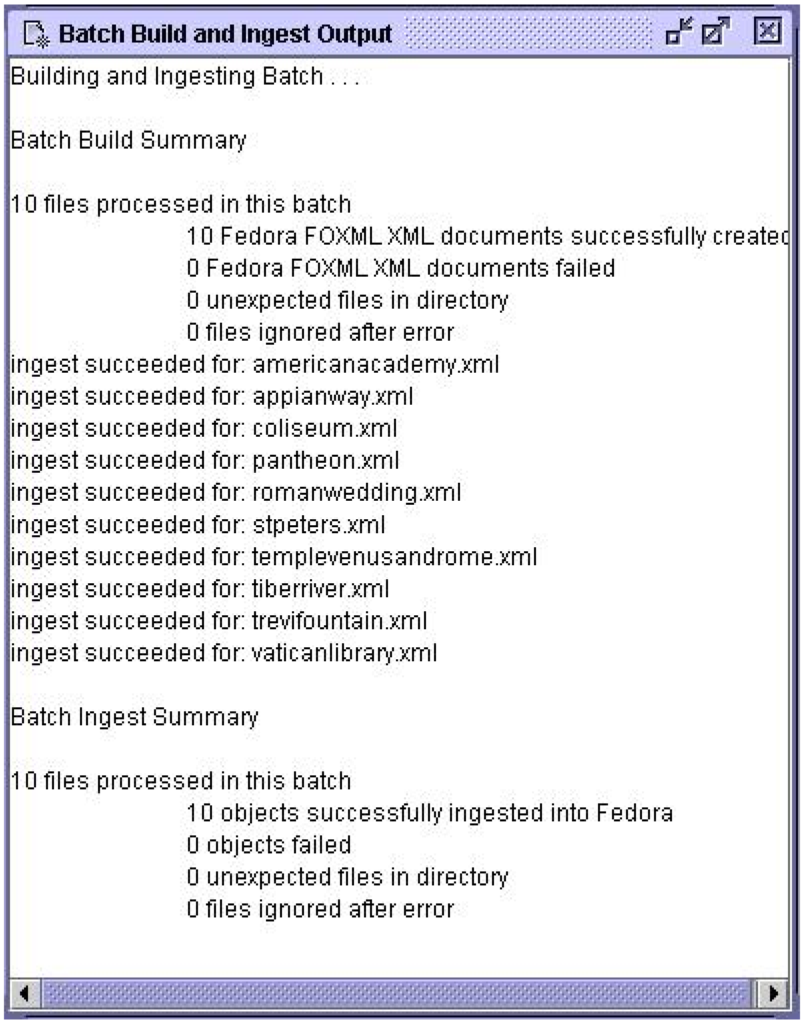

Batch Build and Ingest
Build and Ingest Criteria
Object template (input file)
browse...
XML specs (input directory)
browse...
Fedora objects (output directory)
browse...
object processing map (output file)
● xml ○ text
browse...
Build and Ingest this batch

Then click the *Build and Ingest this batch* button to build the batch of Fedora METS XML documents and then ingest them into Fedora. The format of the built objects is determined by the format of the template. A confirmation dialog will open requesting confirmation of the object template selected. Clicking *Ye* will continue the batch build. Clicking *No* or *Cancel* will return the user to the *Batch Build and Ingest* window.


Generated Object Format

Based on template, generated objects will be in FOXML format. Is this correct?

A second (*output-only*) window will open to show progress. You can build and ingest multiple different batches before closing the *Batch Build and Ingest* window.



```
Building and Ingesting Batch . . .

Batch Build Summary

10 files processed in this batch
                    10 Fedora FOXML XML documents successfully createc
                    0 Fedora FOXML XML documents failed
                    0 unexpected files in directory
                    0 files ignored after error
ingest succeeded for: americanacademy.xml
ingest succeeded for: appianway.xml
ingest succeeded for: coliseum.xml
ingest succeeded for: pantheon.xml
ingest succeeded for: romanwedding.xml
ingest succeeded for: stpeters.xml
ingest succeeded for: templevenusandrome.xml
ingest succeeded for: tiberriver.xml
ingest succeeded for: trevifountain.xml
ingest succeeded for: vaticanlibrary.xml

Batch Ingest Summary

10 files processed in this batch
                    10 objects successfully ingested into Fedora
                    0 objects failed
                    0 unexpected files in directory
                    0 files ignored after error
```

There is then no need to separately ingest the created batch; no directories or files are deleted by Fedora Administrator. Setup and cleanup of the files in the batch must be done by you using standard operating systems facilities. Fedora Administrator does not itself validate on *Batch Build*, but batch ingest into Fedora does. The batch fails on the first individual object ingest failure. Fedora will not ingest a *METS* file whose **METS:xmldata** elements are empty or contain non-tagged character data.

To Demo

You can use files and subdirectories of directory **client/demo/batch-demo**, relative to your **FEDORA_HOME** environment variable. (When you create your own batches, the needed directories and files can be anywhere in the file space of the system on which you are running Fedora Administrator or command-line *BatchTool*.) If you have ingested these objects before, either in this *Build and Ingest Batch* mode or in separate sequential *Build Batch* and *Ingest Batch* modes, you will first need to edit **OBJIDs** in the *object-spec* files, or to remove the corresponding objects from your Fedora repository. Use file **mets-template.xml** for *METS template* (*input file*) if you want to create Fedora *METS* objects. Use file **foxml-template.xml** for *FOXML template* (*input file*) if you want to create Fedora *FOXML* objects. Use subdirectory **object-specifics** for *XML specs* (*input directory*); this is a directory holding (all and only) per-object data.Use subdirectory **objects** for *built objects* (*output directory*); this is a directory to hold (all and only) Fedora object files built by Fedora Administrator.

Specify a file path of your choice for *object processing map* (*output file*); this is a file which maps *object-specs* through objects built and on to PIDs assigned. See the section on *object processing maps*, elsewhere in this documentation. Unlike separate *Batch Build* and *Batch Ingest* modes, the complete triple is reported in this *Batch Build and Ingest* mode.Optionally select the output format for *object processing map*, either **xml** or **text** (**xml** is the default format).

Object Processing Map

The *object-processing-map* file has one of the following formats, depending on the choice of *xml* or *text* in Fedora Administrator. *Batch Build* processing results in an *object processing map* whose individual maps have only **path2spec** and **path2object** attributes or fields. Batch ingest processing results in an object processing map whose individual maps have only **path2object** and **pid** attributes or fields. *Batch build and Ingest* processing results in an *object processing map* whose individual maps have all three **path2spec**, **path2object** and **pid** attributes or fields.

XML Format

```
<object-processing-map>
  <map
    path2spec="/mellon/dist/client/demo/batch-demo/object-specifics/americanacademy.xml"
    path2object=" /mellon/dist/client/demo/batch-demo/objects/americanacademy.xml"
    pid="demo:3010" />
    . . .
  <map
    path2spec="/mellon/dist/client/demo/batch-demo/object-specifics/vaticanlibrary.xml"
    path2object="/mellon/dist/client/demo/batch-demo/objects/vaticanlibrary.xml"
    pid="demo:3019" />
</object-processing-map>
```

Text Format

The field separator is tab; relative paths are used for practical illustration.

```
object-specifics/americanacademy.xml objects/americanacademy.xml demo:3010
. . .
object-specifics/vaticanlibrary.xml objects/vaticanlibrary.xml demo:3019
```

Object-Specifics

Object-specifics are coded in XML files. These data include: object ID, label, and comment; datastream and object metadata and accompanying label; datastream URLs, titles, and labels. Prior to Fedora 2.1, there was no formal *batchMerge* schema for object-specific files and the schema was implied based on the example object-specific files. The new *batchMerge.xsd* schema provides a more formal definition that conforms to the pre-Fedora 2.1 object-specific examples and also provides some new extensions for attributes introduced with *FOXML*. The new schema can be found at: http://www.fedora.info/definitions/1/0/api/batchMerge.xsd.

Where possible, attribute names on elements are the same as in the Fedora *METS* or *FOXML* schema, and so correspond to like-named attributes in the Fedora *METS* or *FOXML* template. How these map is described below and by running the demo and viewing the results for one of the objects. Any individual substitution is optional. When absent as a substitution, the value in the template will be used for the resulting Fedora *METS* or *FOXML* object. (Demo template and *object-specific* contents are chosen instructively to highlight substitutions made.) Datastream URLs will generally be specific to an object; practice will show which other substitutions are generally made. All *non-METS* and *non-FOXML* namespaces used in your own metadata must be declared, as in **xmlns:uvalibadmin** in the demo. The *metadata* element is for backward compatibility with pre-Fedora 2.1 *object-specific* file formats where inline XML metadata was treated separately from other types of datastreams. It is recommended that users use the new extended *datastream* element for all types of datastreams. The *metadata* element may be deprecated in a future release of the batchMerge schema.

When working with a *METS* template object:

Datastream IDs here map to those found in the Fedora **METS:fileGrp** element (the nested, not the nesting, one). The associated **xlink:href** and **xlink:title** attributes are substituted into the Fedora **METS:Flocat** element, which is nested within that Fedora **METS:fileGrp** element. Datastream labels substitute instead into **METS:structMap.**

When working with a *FOXML* template object:

Datastream IDs here map to the **ID** attribute found in the Fedora **foxml:datastream *element. The associated *xlink:href** attribute maps to the **REF** attribute found in the Fedora **foxml:contentLocation** element. The **xlink:title** attribute is for backward compatibility. Both datastream **LABEL** and **xlink:title** attributes map to the **LABEL** attribute found in the Fedora **foxml:datastreamVersion** element. It is recommended that you use the **LABEL** attribute instead of the **xlink:title** attribute when referring to the label of a datastream.

Case matters in attribute and element names.Fedora will retain as PIDs only **OBJIDs** whose prefixes are included in the fedora.fcfg file retainPids parameter (e.g., test, demo, etc.). Other **OBJIDs** will be replaced by Fedora-generated PIDs.*object-specs* in a given batch should meet the structural requirements of that batch's template: same number and tagging of datastreams, same number and tagging of metadata elements. Since substitutions are optional, individual *object-specs* cannot have "missing" data: the resulting object simply retains the template's value. Neither can *object-specs* have "extra" data: the resulting object simply lacks the *object-spec's* data - because the template isn't designed to use it. In either case, the batch goes on.

The following *object-spec* fragment from *americanacademy.xml* illustrates some of this.

```
<?xml version="1.0" encoding="utf-8"?>
<input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xlink="http://www.w3.org/TR/xlink"
       xsi:schemaLocation="http://www.fedora.info/definitions/ http://www.fedora.info/definitions/1/0/api
/batchMerge.xsd"
       xmlns="http://www.fedora.info/definitions/"
       OBJID="demo:3010" LABEL="American Academy">
  <datastreams>
    . . .
    <datastream ID="RIGHTS1">
    <!-- *** TESTING: SUBSTITUTING METADATA FOR RIGHTS1 *** -->
      <xmlContent>
        <uvalibadmin:admin xmlns:uvalibadmin="http://dl.lib.virginia.edu/bin/dtd/admin/admin.dtd">
          <uvalibadmin:adminrights>
            <uvalibadmin:policy>
              <uvalibadmin:access>unrestricted</uvalibadmin:access>
              <uvalibadmin:use>educational</uvalibadmin:use>
            </uvalibadmin:policy>
          </uvalibadmin:adminrights>
        </uvalibadmin:admin>
      </xmlContent>
    </datastream>
      . . .
      other metadata datastreams
      . . .
      <datastream ID="DS1"
                  xlink:href="http://www.fedora.info/demo/batch-demo/thumb/americanacademy.jpg"
                  LABEL="*** TESTING:  SUBSTITUTING LABEL FOR DS1 ***"/>
      <datastream ID="DS2"
                  xlink:href="http://www.fedora.info/demo/batch-demo/medium/americanacademy.jpg"
                  xlink:title="*** TESTING:  SUBSTITUTING XLINK:TITLE FOR DS2 ***"/>
      <datastream ID="DS3" xlink:href="http://www.fedora.info/demo/batch-demo/high/americanacademy.jpg"/>
      <datastream ID="DS4" xlink:href="http://www.fedora.info/demo/batch-demo/very-high/americanacademy.jpg"/>
  </datastreams>
</input>
```

Progress Report File

Fedora Administrator already provides a progress report of each use of a batch tool, written to a GUI window, to provide user feedback. Additionally, this progress report is now written to a text file, to provide a permanent record. Note that this progress report is not especially suited and is not intended for further processing by another computer program. Use the "object processing map", a different output file already provided, for such machine processing. The progress report file is written to the same directory as the object processing map. The name of any instance of these new files includes the time when it is written, e.g., `20031203-123201-365.txt`. The final group of numerals would serve to differentiate report files written, oddly but possibly, at the same second, by 2 instances of the GUI client running on the same machine.

The following description tells what is recorded in this new file and how its directory is chosen in giving the location of the object processing map.The batch tools are available through the Tools menu, under Batch, and serve to provide: Build Batch, Build and Ingest Batch, or Ingest Batch.After selecting one of these tools, a dialog box opens for user input of tool parameters. This dialog box is titled "Batch Build", "Batch Build and Ingest", or "Batch Ingest", depending on which tool is chosen. For each tool, one of the required parameters is the path to the "object processing map" (an output file), which records the tool's processing in a form amendable to later input to another program. This path is specified in a usual file dialog, including its parent directory. This is the parent directory, also, into which the new processing report file is written.

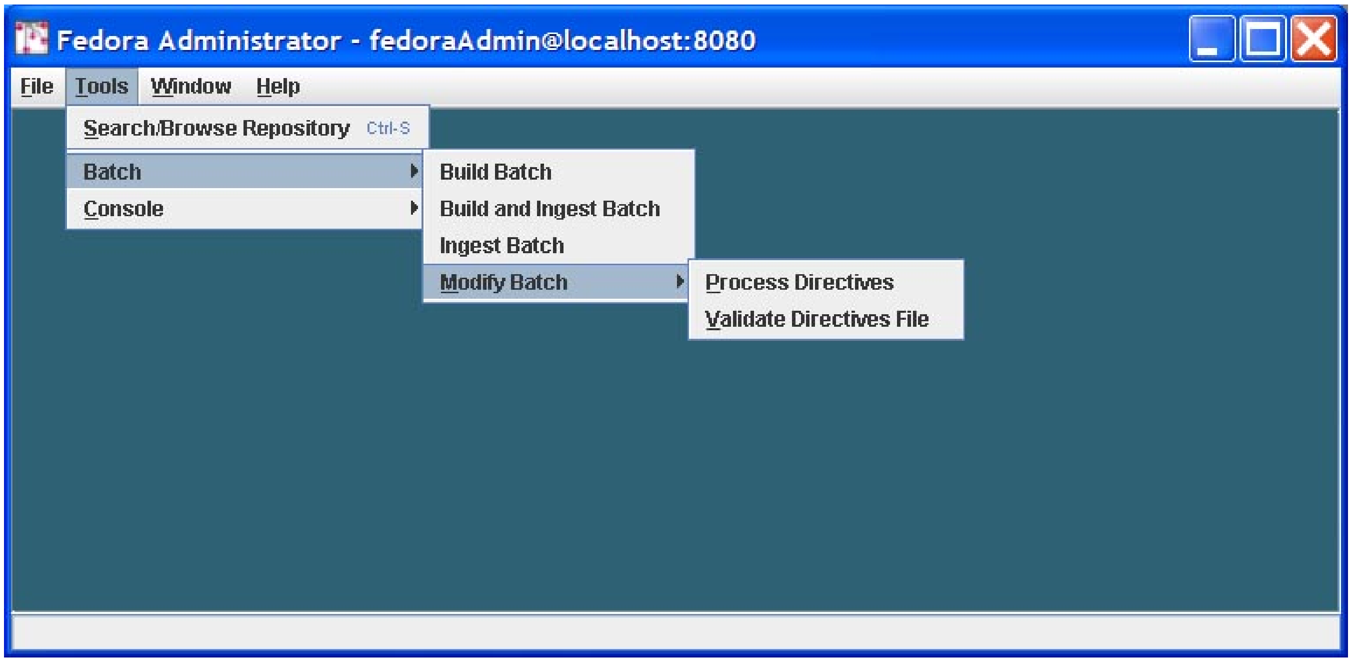
The contents of this new file is simply the contents of the respective output window of the GUI client, one of: "Batch Build Output", "Batch Build and Ingest Output", or "Batch Ingest Output".

Batch Modify

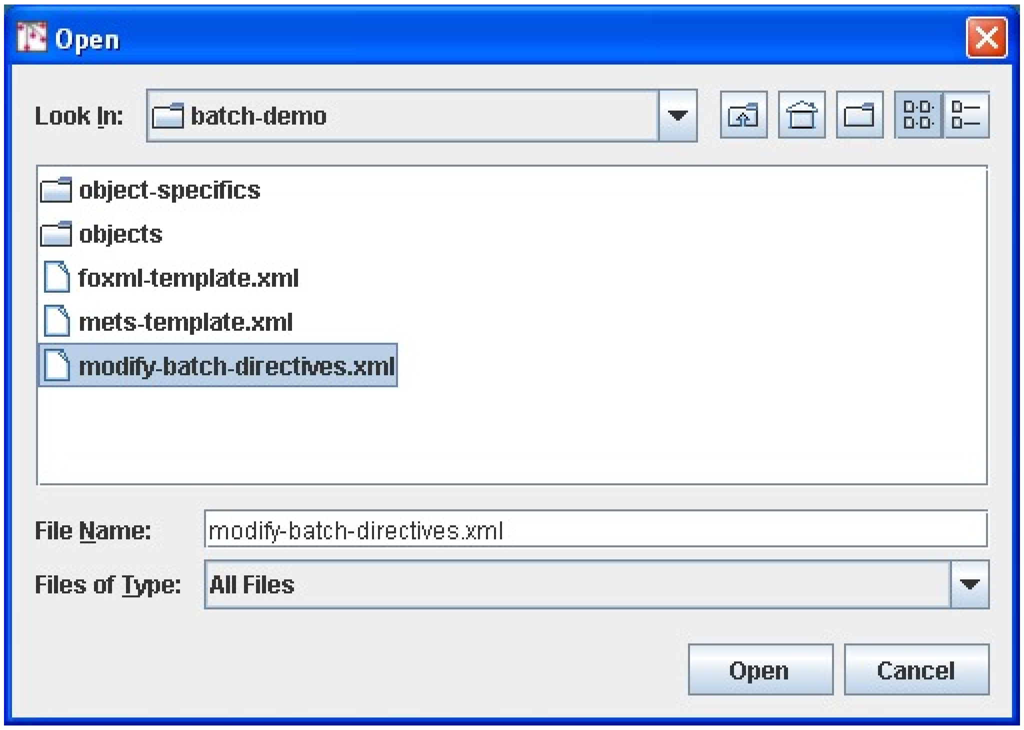
The Batch Modify Utility is an application that enables the modification of objects in batch mode. It is designed for use by repository administrators and is available under the Tools menu of the Administrator GUI client. It can also be invoked as a command-line utility using either the fedora-modify.bat (Windows) or fedora-modify.sh (Unix) scripts located in the distribution client/bin directory. The basic design of the utility is to process an xml input file containing modify directives and then process each directive in sequence using the methods of API-M. The format of the directives file is specified by an xml schema name batchModify.xsd. The schema is available in the local distribution in the tomcat ROOT webapp and is also available from the Fedora website at http://www.fedora.info/definitions/1/0/api/batchModify.xsd. Each modify directive mirrors the capability of the corresponding API-M directives.

Process Directives

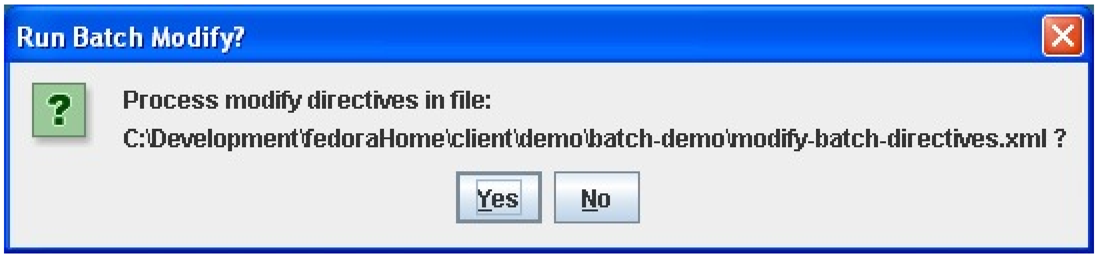
The Batch Modify Utility is accessed through the Tools menu of the Administrator GUI client under the submenu heading of Batch/Modify Batch.



The Modify Batch menu contains two sub-items named Process Directives and Validate Directives File. The Process Directives menu item is used to begin processing a valid set of xml directives in a modify directives file. The Validate Directives File item is used to parse an existing batch modify directives file to insure that it conforms to the Fedora Batch Modify XML schema.Selecting the Process Directives item will prompt for the location of the directives file.

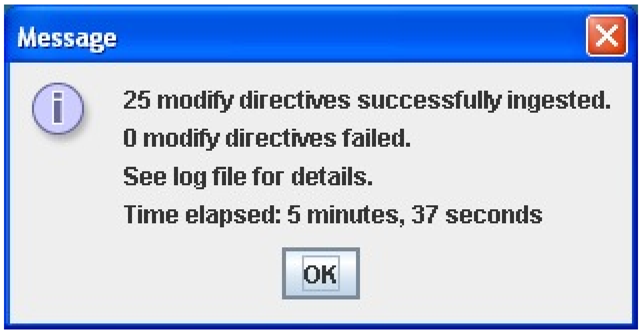


After selecting a directives file, you will be asked to confirm the choice.


Run Batch Modify?

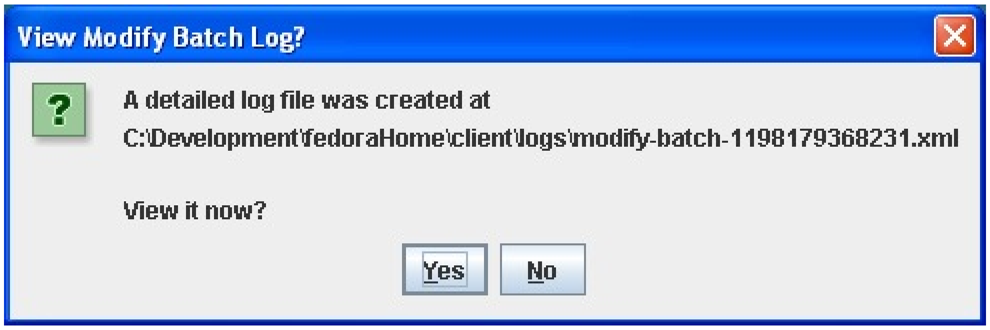
Process modify directives in file:
C:\Development\fedoraHome\client\demo\batch-demo\modify-batch-directives.xml ?

Upon clicking the yes button, the directives file will be processed and a log file generated that details the number of directives processed and any errors that may have occurred.


Message

25 modify directives successfully ingested.
0 modify directives failed.
See log file for details.
Time elapsed: 5 minutes, 37 seconds

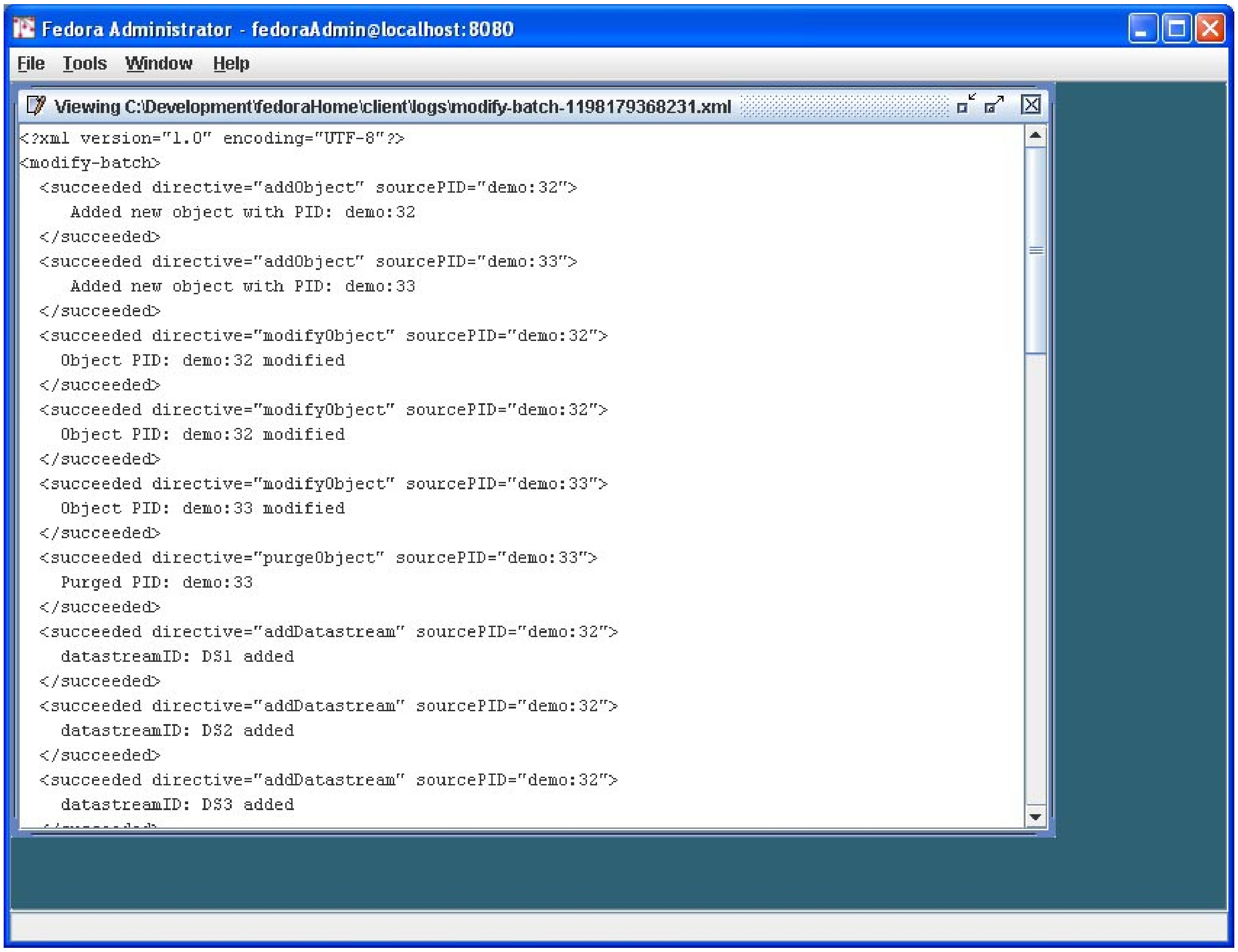
If there were any errors, you can peruse the log file to ascertain more information about what caused the directive to fail.


View Modify Batch Log?

A detailed log file was created at
C:\Development\fedoraHome\client\logs\modify-batch-1198179368231.xml

View it now?

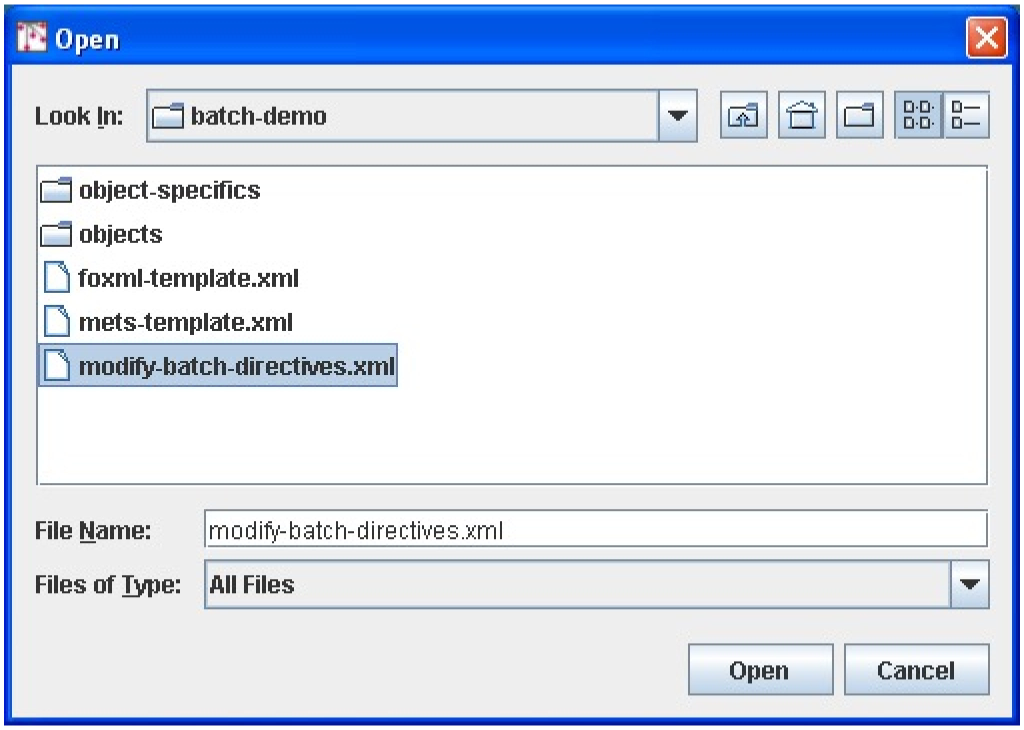
The log file consists of an xml file containing entries for each directive that was processed.


Fedora Administrator - fedoraAdmin@localhost:8080
File Tools Window Help
Viewing C:\Development\fedoraHome\client\logs\modify-batch-1198179368231.xml

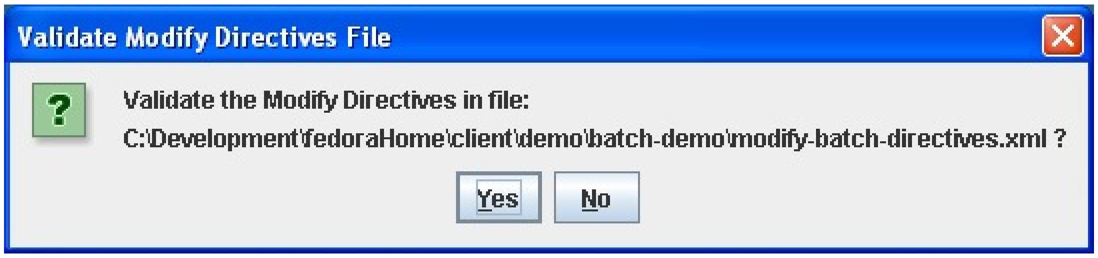
```
<?xml version="1.0" encoding="UTF-8"?>
<modify-batch>
  <succeeded directive="addObject" sourcePID="demo:32">
     Added new object with PID: demo:32
  </succeeded>
  <succeeded directive="addObject" sourcePID="demo:33">
     Added new object with PID: demo:33
  </succeeded>
  <succeeded directive="modifyObject" sourcePID="demo:32">
    Object PID: demo:32 modified
  </succeeded>
  <succeeded directive="modifyObject" sourcePID="demo:32">
    Object PID: demo:32 modified
  </succeeded>
  <succeeded directive="modifyObject" sourcePID="demo:33">
    Object PID: demo:33 modified
  </succeeded>
  <succeeded directive="purgeObject" sourcePID="demo:33">
    Purged PID: demo:33
  </succeeded>
  <succeeded directive="addDatastream" sourcePID="demo:32">
    datastreamID: DS1 added
  </succeeded>
  <succeeded directive="addDatastream" sourcePID="demo:32">
    datastreamID: DS2 added
  </succeeded>
  <succeeded directive="addDatastream" sourcePID="demo:32">
    datastreamID: DS3 added
```

Validate Directives

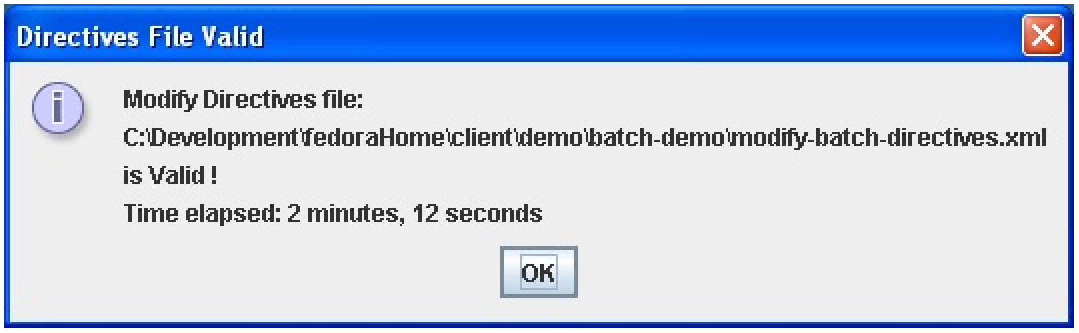
The Validate Directives menu item is used to verify that an existing directives file is valid before attempting to process the file. This can be useful when the directive files were created using a non-xml editor that was unable to validate the file as it was created. The steps for validating a directive file are similar to that for processing. First you are prompted for the location of the directives file to be processed.



Confirming the file location initiates the parsing process and the results are displayed when parsing is completed.



Validate the Modify Directives in file:
C:\Development\fedoraHome\client\demo\batch-demo\modify-batch-directives.xml ?



Modify Directives file:
C:\Development\fedoraHome\client\demo\batch-demo\modify-batch-directives.xml
is Valid !
Time elapsed: 2 minutes, 12 seconds

To Demo

There is a sample batch modify directives file included in the distribution located in the dist/client/demo/batch-demo directory named modify-batch-directives.xml. This sample file creates a new object and then performs various additions, deletions, and modifications to the object's components through s series of directives. For reference the the sample directives file is included below.

Sample Modify Directives File

```
<?xml version="1.0" encoding="utf-8"?>
<!--  *******************************************************************  -->
<!--  This is a sample modify directives file that performs a variety of  -->
<!--  modfications on the datastreams of two new demo  -->
<!--  objects (demo:32 and demo:33) that the script will create. In general,  -->
<!--  one should always validate a modify directives file against the  -->
<!--  modifyBatch.xsd schema prior to processing to catch any syntax errors.  -->
```

```
<!--  Processing will halt if the directives file is invalid and a log file  -->
<!--  is generated summarizing the results of the batch. Refer to the xml  -->
<!--  schema file (modifyBatch.xsd) for details on the syntax of the modify  -->
<!--  file.  -->
<!--  -->
<!--  *********************************************************************  -->
<  fbm:batchModify xmlns:fbm="http://www.fedora.info/definitions/"
                   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xsi:schemaLocation="http://www.fedora.info/definitions/
                                       http://www.fedora.info/definitions/1/0/api/batchModify.xsd">
<!--  *****************************************************************  -->
<!--  Add a new empty object to the repository that has no datastreams  -->
<!--  with PID of demo:32.  -->
<!--  -->
<!--  NOTE: By default each Fedora repository specifies several PID  -->
<!--  namespaces (e.g. "demo")that will be retained at ingest by the  -->
<!--  repository allowing ingested objects to retain the PID specified  -->
<!--  at ingest time. This list of PID namespaces is configurable in  -->
<!--  fedora.fcfg using the retainPids config parameter. In this example,  -->
<!--  demo:32 is specified PID so the generated object will retain this  -->
<!--  PID value after ingestion. Leaving the pid attribute empty  -->
<!--  (i.e., "") will force the repository to assign a PID to the new  -->
<!--  object using the namespace specified by the PidNamespace parameter  -->
<!--  in the fedora.fcf config file.  -->

<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object; leave blank if you want repository to  -->
<!--  assign the pid.  -->
<!--  label - label for the object  -->
<!--  contentModel - content model of the object  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  ******************************************************************  -->
<fbm:addObject pid="demo:32" label="Sample Object Used With Batch Modify Utility"
               contentModel="Object" logMessage="BatchModify - addObject"/>
<!--  *****************************************************************  -->
<!--  Add a second new empty object to the repository that has no  -->
<!--  datastreams with a PID of demo:33.  -->
<!--  ******************************************************************  -->
<fbm:addObject pid="demo:33" label="2nd Sample Object Used With Batch Modify Utility"
               contentModel="Object" logMessage="BatchModify - addObject"/>
<!--  *****************************************************************  -->
<!--  Modify object demo:32 by changing the label on the object.  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  -->
<!--  Optional attributes on directive:  -->
<!--  label - label for the object  -->
<!--  state - state of the object  -->
<!--  ******************************************************************  -->
<fbm:modifyObject pid="demo:32" label="Object Label was changed" logMessage="BatchModify - modifyObject"/>
<!--  *****************************************************************  -->
<!--  Modify object demo:33 by changing its state to deleted.  -->
<!--  ******************************************************************  -->
<fbm:modifyObject pid="demo:33" state="D" logMessage="BatchModify - modifyObject  "/>
<!--  *****************************************************************  -->
<!--  Purge object demo:33 that was just created and modified.  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object to be purged  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  -->
<!--  Optional attributes on directive:  -->
<!--  force - boolean indicating whether to purge the object if any  -->
<!--  critical dependencies exist. A value of true will purge the object  -->
<!--  regardless of any dependencies. A value of false will allow the  -->
<!--  purge only if no dependencies exist. This parameter is currently  -->
<!--  not implemented in the server.  -->
<!--  ******************************************************************  -->
```

```
<fbm:purgeObject pid="demo:33" logMessage="BatchModify - purgeObject"/>
<!--  *****************************************************************  -->
<!--  Add a new datastream to demo:32 object with ID of DS1.  -->
<!--  -->
<!--  Note: A dsID of DS1 is assigned by the server since no dsID is  -->
<!--  specified on the directive and this is the first datastream in  -->
<!--  this object.  -->
<!--  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object  -->
<!--  dsLabel - label fo the datastream  -->
<!--  dsMIME - MIME type of the datastream  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  dsState - state of the object  -->
<!--  dsControlGroupType - control group type of the datastream  -->
<!--  dsLocation - location of the datastream; for XMLMetadataDatastreams  -->
<!--  dsLocation omit dsLocation or leave as blank. For all  -->
<!--  other datastream types, it is required and must  -->
<!--  denote the remote location of the datastream.  -->
<!--  -->
<!--  Optional attributes on directive:  -->
<!--  dsID - ID of the datastream; omit or leave as blank if you want  -->
<!--  repository to assign datastream ID.  -->
<!--  formatURI - URI identifying the format of the datastream  -->
<!--  versionable - boolean indicating whether datastream is versionable  -->
<!--  altIDs - space delimited string of alternated identifiers for the  -->
<!--  datastream. This info is currently maintained in the  -->
<!--  object, but not acted on.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsLabel="Thorny's Coliseum thumbnail jpg image" dsMIME="image/jpeg"
                   dsLocation="http://localhost:8080/demo/simple-image-demo/coliseum-thumb.jpg"
                   dsControlGroupType="E" dsState="A" logMessage="BatchModify - addDatastream"/>
<!--  *****************************************************************  -->
<!--  Add a 2nd datastream to demo:32 object. The new datastream will be  -->
<!--  a medium resolution image and will be assigned a dsID of DS2 since  -->
<!--  no dsID is specified and this is the second datastream for this  -->
<!--  object.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsLabel="Thorny's Coliseum medium jpg image" dsMIME="image/jpeg"
                   dsLocation="http://localhost:8080/demo/simple-image-demo/coliseum-medium.jpg"
                   dsControlGroupType="E" dsState="A" logMessage="BatchModify - addDatastream  "/>
<!--  *****************************************************************  -->
<!--  Add a 3rd datastream to demo:32 object. The new datastream will be  -->
<!--  a high resolution image and will be assigned a dsID of DS3 since  -->
<!--  no dsID is specified and this is the third datastream for this  -->
<!--  object.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsLabel="Thorny's Coliseum high jpg image" dsMIME="image/jpeg"
                   dsLocation ="http://localhost:8080/demo/simple-image-demo/coliseum-high.jpg"
                   dsControlGroupType="M" dsState="A" logMessage="BatchModify - addDatastream"/>
<!--  *****************************************************************  -->
<!--  Add a 4th datastream to demo:32 object. The new datastream will be  -->
<!--  a very high resolution image and will be assigned a dsID of DS4  -->
<!--  since no dsID is specified and this is the fourth datastream for  -->
<!--  this object.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsLabel="Thorny's Coliseum veryhigh jpg image" dsMIME="image/jpeg"
                   dsLocation="http://localhost:8080/demo/simple-image-demo/coliseum-veryhigh.jpg"
                   dsControlGroupType="M" dsState="A" logMessage="BatchModify - addDatastream"/>
<!--  *****************************************************************  -->
<!--  Add a 5th datastream to demo:32 object. The new datastream will be  -->
<!--  a screen size image and will have dsID of SCREEN. This datastream  -->
<!--  will also have values assigned for alternate IDs, formatURI, and  -->
<!--  is declared not to be versionable.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsID="SCREEN" altIDs="AlternateID1 AlternateID2"
                   formatURI="info:fedora/demo/content/JPEG#" versionable="false"
                   dsLabel="Thorny's Coliseum screen size jpg image" dsMIME="image/jpeg"
                   dsLocation="http://localhost:8080/demo/simple-image-demo/coliseum-high.jpg"
                   dsControlGroupType="E" dsState="A" logMessage="BatchModify - addDatastream"/>
```

```
<!--  *****************************************************************  -->
<!--  Add a 6th datastream to demo:32 object. The new datastream will be  -->
<!--  a user-defined inline descriptive metadata datastream and will  -->
<!--  have dsID of DESC.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsID="DESC" dsLabel="Descriptive metadata for Thorny's Coliseum screen size
jpg image"
                   dsMIME="text/xml" dsControlGroupType="X" dsState="A" logMessage="BatchModify - addDatastream"
>
<fbm:xmlData>
<uvalibdesc:desc xmlns:uvalibdesc="http://dl.lib.virginia.edu/bin/dtd/descmeta/descmeta.dtd">
<uvalibdesc:time>
  <uvalibdesc:date type="created" certainty="ca." era="bc">1st century</uvalibdesc:date>
</uvalibdesc:time>
<uvalibdesc:identifier scheme="URN">uva-lib:2</uvalibdesc:identifier>
<uvalibdesc:rights type="use">unrestricted</uvalibdesc:rights>
<uvalibdesc:subject scheme="other" othertype="keyword">Roman Empire</uvalibdesc:subject>
<uvalibdesc:subject scheme="other" othertype="keyword">Roman</uvalibdesc:subject>
<uvalibdesc:subject scheme="other" othertype="keyword">Vespaciano</uvalibdesc:subject>
<uvalibdesc:subject scheme="other" othertype="keyword">Amphitheatrum Flavium</uvalibdesc:subject>
<uvalibdesc:title type="main">Coliseum -- Rome, Italy</uvalibdesc:title>
<uvalibdesc:form>architecture</uvalibdesc:form>
<uvalibdesc:mediatype type="image"><uvalibdesc:form>digital</uvalibdesc:form></uvalibdesc:mediatype>
<uvalibdesc:agent role ="publisher">Alderman Library</uvalibdesc:agent>
<uvalibdesc:covspace>
  <uvalibdesc:geometry>
    <uvalibdesc:point>
      <uvalibdesc:lat>41.54N</uvalibdesc:lat><uvalibdesc:long>12.27E</uvalibdesc:long>
    </uvalibdesc:point>
  </uvalibdesc:geometry>
</uvalibdesc:covspace>
<uvalibdesc:covtime>
  <uvalibdesc:date era="bc">72</uvalibdesc:date><uvalibdesc:date era ="bc">80</uvalibdesc:date>
</uvalibdesc:covtime>
<uvalibdesc:culture>Roman</uvalibdesc:culture>
<uvalibdesc:place type="original"><uvalibdesc:geogname>Italy, Rome</uvalibdesc:geogname></uvalibdesc:place>
</uvalibdesc:desc>
</fbm:xmlData>
</fbm:addDatastream>
<!--  *****************************************************************  -->
<!--  Add a 7th datastream to demo:32 object. The new datastream will be  -->
<!--  duplicate of the screen size image that will be deleted later in  -->
<!--  this set of directives. It will have a dsID of SCREENDUP.  -->
<!--  *****************************************************************  -->
<fbm:addDatastream pid="demo:32" dsID="SCREENDUP" dsLabel="Thorny's Coliseum screen size jpg image" dsMIME="
image/jpeg"
                   dsLocation="http://localhost:8080/demo/simple-image-demo/coliseum-high.jpg"
                   dsControlGroupType="R" dsState="A" logMessage="BatchModify - addDatastream"/>
<!--  *****************************************************************  -->
<!--  Modify datastream DS1 of demo:5 object by changing its label.  -->
<!--  -->
<!--  NOTE: Optional attributes that are omitted indicate that that  -->
<!--  attribute is to remain uncahnged and the original value of of the  -->
<!--  datastream attribute in the object will be preserved. Optional  -->
<!--  attributes that are set to the empty string indicate that the  -->
<!--  atrribute value will be blanked out in the datastream.  -->
<!--  In this example the datastream location and state attributes are  -->
<!--  not modified.  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object  -->
<!--  dsID - ID of the datastream  -->
<!--  dsControlGroupType - control group type of the datastream  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  -->
<!--  Optional attributes on directive:  -->
<!--  dsLabel - label fo the datastream  -->
<!--  dsState - state of the object  -->
<!--  dsLocation - location of the datastream  -->
<!--  dsMIME - MIME type of the datastream  -->
<!--  formatURI - URI identifying the format of the datastream  -->
```

```
<!--  versionable - boolean indicating whether datastream is versionable  -->
<!--  altIDs - space delimited string of alternated identifiers for the  -->
<!--  datastream. This info is currently maintained in the  -->
<!--  object, but not acted on.  -->
<!--  force - boolean indicating whether to purge the object if any  -->
<!--  critical dependencies exist. A value of true will purge  -->
<!--  the object regardless of any dependencies. A value of  -->
<!--  false will allow the purge only if no dependencies exist.  -->
<!--  *****************************************************************  -->
<fbm:modifyDatastream pid="demo:32" dsID="DS1" dsControlGroupType="E"
                      dsLabel="New label for datastream DS1"
                      logMessage="BatchModify - modifyDatastream"/>
<!--  *****************************************************************  -->
<!--  Modify datastream SCREEN of demo:32 object by changing its label  -->
<!--  and datastream location so that it points to the same content as  -->
<!--  that of datastream DS3 which is a high-res image.  -->
<!--  -->
<!--  NOTE: Optional attributes that are omitted indicate that that  -->
<!--  attribute is to remain uncahnged and the original value of of the  -->
<!--  datastream attribute in the object will be preserved. Optional  -->
<!--  attributes that are set to the empty string indicate that the  -->
<!--  attribute value will be blanked out in the datastream.  -->
<!--  *****************************************************************  -->
<fbm:modifyDatastream pid="demo:32" dsID="SCREEN" dsControlGroupType="E"
                      dsLabel="Changed label for datastream SCREEN"
                      dsLocation="http://localhost:8080/fedora/get/demo:5/DS4"
                      logMessage="BatchModify - modifyDatastream"/>
<!--  *****************************************************************  -->
<!--  Modify datastream DC of demo:32 object by changing its label and  -->
<!--  its xml content so that its content is replaced by the block of  -->
<!--  xml below.  -->
<!--  *****************************************************************  -->
<fbm:modifyDatastream pid="demo:32" dsID="DC" dsControlGroupType="X"
                      dsLabel="New label for DC datastream"
                      logMessage="BatchModify - modifyDatastream">
<fbm:xmlData>
  <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
             xmlns:dc="http://purl.org/dc/elements/1.1/"
             xmlns:my="http//foo.bar.none">
    <dc:title>Coliseum in Rome</dc:title>
    <dc:creator>Thornton Staples</dc:creator><dc:subject>Architecture, Roman</dc:subject>
    <dc:description>Image of Coliseum in Rome</dc:description>
    <dc:publisher>University of Virginia Library</dc:publisher>
    <dc:format>image/jpeg</dc:format>
    <dc:identifier>demo:5</dc:identifier>
    <my:this>some text</my:this>
    <my:that>some more text</my:that>
    <my:other>even more text</my:other>
  </oai_dc:dc>
</fbm:xmlData>
</fbm:modifyDatastream>
<!--  *****************************************************************  -->
<!--  Modify datastream DESC of demo:32 object by changing its label.  -->
<!--  Also change its formatURI and set versionable to false.  -->
<!--  *****************************************************************  -->
<fbm:modifyDatastream pid="demo:32" dsID="DESC" dsControlGroupType="X"
                      dsLabel="New label for DC datastream" formatURI="info:fedora/new/formatURI"
                      versionable="false"logMessage="BatchModify - modifyDatastream"/>
<!--  *****************************************************************  -->
<!--  Purge datastream SCREENDUP from object demo:32  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object to be purged  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  -->
<!--  Optional attributes on directive:  -->
<!--  force - boolean indicating whether to purge the object if any  -->
<!--  critical dependencies exist. A value of true will purge  -->
<!--  the object regardless of any dependencies. A value of  -->
<!--  false will allow the purge only if no dependencies exist.  -->
<!--  purge only if no dependencies exist. This parameter is  -->
```

```
<!--  currently not implemented in the server.  -->
<!--  ******************************************************************  -->
<fbm:purgeDatastream pid="demo:32" dsID="SCREENDUP" logMessage="BatchModify - purgeDatastream"/>
<!--  ******************************************************************  -->
<!--  Change the state of datastream DS1 to deleted  -->
<!--  -->
<!--  Required attributes on directive:  -->
<!--  pid - PID of the object  -->
<!--  dsID - ID of the datastream  -->
<!--  dsState - new value for state of the datastream  -->
<!--  logMessage - message to be written in audit trail record  -->
<!--  ******************************************************************  -->
<fbm:setDatastreamState pid="demo:32" dsID="DS1" dsState="D" logMessage ="BatchModify - setDatastreamState"/>
<!--  ******************************************************************  -->
<!--  Change the state of datastream DESC to inactive  -->
<!--  ******************************************************************  -->
<fbm:setDatastreamState pid="demo:32" dsID="DESC" dsState="I" logMessage="BatchModify - setDatastreamState"/>
</fbm:batchModify>
```

Command Line Alternatives

Rather than using the Fedora Administrator UI, you can build, ingest, and modify batches of Fedora object with the command-line utilities included with Fedora.

Please see relevant usage instructions at Client Command Line Utilitiesfor the following scripts:

- fedora-batch-build
- fedora-batch-ingest
- fedora-batch-buildingest
- fedora-modify