

# Simple Archive Format Packager

Note

The project home for this project is: <https://github.com/DSpace-Labs/SAFBuilder>

SAF Packager is becoming outdated, other projects exist

After the core developer left, the SAF Packager / SAFBuilder has not been maintained in recent years. But a number of newer, similar programs exist from other community members:

- <https://github.com/jcreel/SAFCreator>
- <https://github.com/isido/saf-archiver>
- <https://github.com/lib-uoguelph-ca/dspace-csv-archive>

The input for a command-line batch ingest of materials to DSpace is well documented, and is called "Simple Archive Format", however there needs to be a tool that easily facilitates creating a Simple Archive Format package. The use case satisfied with the Simple Archive Format Packager is that someone has a spreadsheet filled with metadata as well as content files that are eventually destined for repository ingest.








Thus the input to the Simple Archive Format Packager is a spreadsheet (.csv) that has the following columns:

- **filename** of the content file(s)
- **namespace.element.qualifier** metadata for the item. Examples would be: dc.description or dc.contributor.author

Further, dates need to be in ISO-8601 format in order to be properly recognized. And for any column that has multiple values, you can separate each entry with a double-pipe "|". For example, for multiple files just set "filename" to "file1.pdf||file2.pdf||file3.pdf". Similarly, multiple "dc.subject" values can be separated by "|" as shown in the below example.

|   | A                                | B   | C              | D   |
|---|----------------------------------|---|----------------|---|
| 1 | filename                         | dc.title  | dc.date.issued | dc.subject  |
| 2 | OCA_GlennArchives_G052-02-1.jpg  | John H. Glenn, Sr. in World War I uniform                         | 1918           | John H. Glenn, Sr.  World War I  Glenn Family               |
| 3 | OCA_GlennArchives_G052-06-1.jpg  | Clara Sproat Glenn  | 1920           | Clara Sproat Glenn  Glenn Family                            |
| 4 | OCA_GlennArchives_G052-06-2.jpg  | Ice Skaters at Muskingum College                                  | 1916           | Clara Sproat Glenn  Glenn Family  New Concord  Muskingum    |
| 5 | OCA_GlennArchives_G052-06-3x.jpg | 1907 Laughlin family reunion                                      | 1907           | Clara Sproat Glenn  |
| 6 | OCA_GlennArchives_G052-07-1.jpg  | John H. Glenn, Sr. in World War I uniform with Clara Sproat Glenn | 1918           | John H. Glenn, Sr.  Clara Sproat Glenn  World War I  Family |

While you are preparing the batch load, you have a directory containing a spreadsheet filled with metadata and content files.

|   |                                  |                       |
|---|----------------------------------|-----------------------|
|  | Glenn-Batch-Load-In-Progress     | 537 items folder      |
|  | GlennPhotoGallery-batch.csv      | 295.1 KB CSV document |
|  | OCA_GlennArchives_G052-02-1.jpg  | 87.5 KB JPEG Image    |
|  | OCA_GlennArchives_G052-06-1.jpg  | 57.2 KB JPEG Image    |
|  | OCA_GlennArchives_G052-06-2.jpg  | 70.0 KB JPEG Image    |
|  | OCA_GlennArchives_G052-06-3x.jpg | 229.8 KB JPEG Image   |
|  | OCA_GlennArchives_G052-07-1.jpg  | 102.8 KB JPEG Image   |

## Obtaining, Compiling, and Running SAFBuilder

The SAFBuilder project resides on GitHub. Please refer to the project instructions for how to install and run it. Its requires Java JDK 7+, and runs from the terminal / command prompt.

The `./safbuilder.sh` command with no arguments will show the help screen.

```
Recompiling SAFBuilder, just a moment...

usage: SAFBuilder

-c,--csv <arg>  Filename with path of the CSV spreadsheet. This must be in the same directory as the content files

-h,--help      Display the Help

-m,--manifest   Initialize a spreadsheet, a manifest listing all of the files in the directory, you must specify a CSV for -c














-z,--zip       (optional) ZIP the output
```

There is sample data included with the tool to give an idea of how to use this.

To run the tool over the sample data:

```
./safbuilder.sh -c /home/dspace/SAFBuilder/src/sample_data/AAA_batch-metadata.csv
```

This creates the SimpleArchiveFormat directory inside of the directory specified, along with subdirectories, content files, metadata files that is ready to import into DSpace.

|   |                              |
|---|------------------------------|
| ▼  SimpleArchiveFormat             | 537 items folder             |
| ▼  item_1                          | 3 items folder               |
|  contents                         | 32 bytes plain text document |
|  dublin_core.xml                 | 1.1 kB XML document          |
|  OCA_GlennArchives_G052-02-1.jpg | 89.6 kB JPEG Image           |
| ▼  item_2                        | 3 items folder               |
|  contents                        | 32 bytes plain text document |
|  dublin_core.xml                 | 1.0 kB XML document          |
|  OCA_GlennArchives_G052-06-1.jpg | 58.6 kB JPEG Image           |
| ▼  item_3                        | 3 items folder               |
|  contents                        | 32 bytes plain text document |
|  dublin_core.xml                 | 1.2 kB XML document          |
|  OCA_GlennArchives_G052-06-2.jpg | 71.7 kB JPEG Image           |

This is then immediately ready to be batch imported into DSpace. If you created a ZIP file of this, that can be imported to DSpace using Batch Import UI. An example of DSpace command line import is.

```
sudo /dspace/bin/dspace import -a
-e peter@longsight.com
-c 1811/49710
-s /home/dspace/SAFBuilder/src/sample_data/SimpleArchiveFormat/
-m /home/dspace/SAFBuilder/src/sample_data/batch1.map
```

## Further Work

This packager works as a stand-alone tool, and requires knowledge of Java to be able to run. Thus satisfying the initial need to be able to package many items to be batch loaded into DSpace, using DSpace's launcher item-import. So the remaining goal of this project is to streamline the process of batch loading materials into DSpace.

Possibilities include:

- refactoring so that it can become a [Packager Plugin](#). Packager plugins allow you to implement a way for DSpace to accept an input package (containing content files, manifest, and metadata) that then creates DSpace items.
- creating a client GUI for the desktop.
- Dedicated web service