



## **Dataverse Documentation**

*Release 4.17*

**Dataverse Team**

Oct 23, 2019



# CONTENTS

<b>1</b>	<b>User Guide</b>	<b>3</b>
1.1	Account Creation + Management	3
1.2	Finding and Using Data	8
1.3	Dataverse Management	14
1.4	Dataset + File Management	24
1.5	Tabular Data File Ingest	42
1.6	Data Exploration Guide	49
1.7	Appendix	58
<b>2</b>	<b>Admin Guide</b>	<b>59</b>
2.1	Dashboard	59
2.2	External Tools	60
2.3	Managing Harvesting Clients	63
2.4	Managing Harvesting Server and Sets	64
2.5	Metadata Customization	66
2.6	Metadata Export	76
2.7	Dataverse Application Timers	77
2.8	Make Data Count	78
2.9	Integrations	82
2.10	Geoconnect and WorldMap	85
2.11	User Administration	87
2.12	Managing Datasets and Dataverses	89
2.13	Solr Search Index	91
2.14	IP Groups	93
2.15	Monitoring	95
2.16	Reporting Tools	98
2.17	Maintenance	98
2.18	Backups	98
2.19	Troubleshooting	99
<b>3</b>	<b>API Guide</b>	<b>103</b>
3.1	Introduction	103
3.2	Getting Started with APIs	108
3.3	API Tokens and Authentication	111
3.4	Search API	112
3.5	Data Access API	118
3.6	Native API	123
3.7	Metrics API	165
3.8	SWORD API	167
3.9	Client Libraries	174

3.10	Building External Tools	175
3.11	Apps	180
3.12	Frequently Asked Questions	183
<b>4</b>	<b>Installation Guide</b>	<b>187</b>
4.1	Introduction	187
4.2	Preparation	188
4.3	Prerequisites	192
4.4	Installation	202
4.5	Configuration	208
4.6	Upgrading	251
4.7	TwoRavens	251
4.8	Geoconnect	265
4.9	Shibboleth	265
4.10	OAuth Login: ORCID, GitHub, Google	279
4.11	External Tools	282
4.12	Advanced Installation	283
<b>5</b>	<b>Developer Guide</b>	<b>285</b>
5.1	Introduction	285
5.2	Development Environment	287
5.3	Windows Development	291
5.4	Tips	294
5.5	Troubleshooting	299
5.6	Version Control	302
5.7	SQL Upgrade Scripts	305
5.8	Testing	306
5.9	Writing Documentation	318
5.10	Dependency Management	321
5.11	Debugging	326
5.12	Coding Style	326
5.13	Deployment	329
5.14	Docker, Kubernetes, and OpenShift	332
5.15	Making Releases	341
5.16	Tools	343
5.17	Universal Numerical Fingerprint (UNF)	349
5.18	Make Data Count	356
5.19	Shibboleth and OAuth	359
5.20	Geospatial Data	360
5.21	SELinux	364
5.22	Big Data Support	368
5.23	Workflows	374
<b>6</b>	<b>Style Guide</b>	<b>377</b>
6.1	Foundations	377
6.2	Patterns	381
<b>7</b>	<b>How the Guides Are Organized</b>	<b>389</b>
<b>8</b>	<b>Other Resources</b>	<b>391</b>
<b>9</b>	<b>Indices and Tables</b>	<b>393</b>

These documentation guides are for the 4.17 version of Dataverse. To find guides belonging to previous versions, `guides_versions` has a list of all available versions.



**Contents:**

## 1.1 Account Creation + Management

**Contents:**

- *Account Information*
  - *Account Log In Options*
  - *Create Account*
  - *Edit Account*
  - *Convert Account*
  - *Reset Account Password*
- *Remote Authentication*
  - *Institutional Log In*
    - \* *Create a Dataverse account using Institutional Log In*
    - \* *Convert your Dataverse account to use your Institutional Log In*
    - \* *Convert your Dataverse account away from your Institutional Log In*
  - *ORCID Log In*
    - \* *Create a Dataverse account using ORCID*
    - \* *Convert your Dataverse account to use ORCID for log in*
    - \* *Convert your Dataverse account away from ORCID for log in*
  - *GitHub and Google Log In*
- *My Data*
- *Notifications*
- *API Token*
  - *What APIs Are and Why They Are Useful*
  - *How Your API Token Is Like a Password*

- *How to Create Your API Token*
- *How to Recreate Your API Token*
- *Additional Information about API Tokens and Dataverse APIs*

### 1.1.1 Account Information

As a registered user, you can:

- Create your own dataverse, if permitted, and customize it.
- Add datasets to dataverses, if permitted.
- Contribute to existing datasets, if permitted.
- Request access to restricted files, if permitted.

### Account Log In Options

Dataverse has been configured for one or more of the following log in options:

- Username/Email and Password
- Institutional Log In
- ORCID
- GitHub
- Google

Please note that once you create your Dataverse account, it will be associated with only one of the log in options above.

The Institutional Log In, ORCID, GitHub, and Google options are described in more detail below under “Remote Authentication.”

### Create Account

To create a Dataverse account with the Username/Email log in option, use the “Sign Up” page. Fill out the fields, and then click the ‘Create Account’ button. Please note that the Username field does not support email addresses but will allow the following characters: a-Z, 0-9, \_ (underscores), - (hyphens), and . (periods).

To create a Dataverse account associated with the log in option for your institution, ORCID, GitHub, or Google, use the “Log In” page and select one of the authentication providers. See the “Remote Authentication” section below for details.

### Edit Account

1. To edit your account after you have logged in, click on your account name in the header on the right hand side and click on Account Information.
2. On the top right of your account page, click on the “Edit Account” button and from there you can select to edit either your Account Information or your Account Password.
3. Select “Save Changes” when you are done.



Please note that you cannot edit your account information within Dataverse if you use the Institutional Log In option. Instead, you should contact your institution to change your name, email, etc. Once the change is made by your institution, it will be reflected in Dataverse the next time you log in. Users of the Institutional Log In option are not required to verify their email address because the institution providing the email address is trusted.

## **Convert Account**

If more than one log in option is available, you can convert from one to another and use the new option from now on.

If you are converting from the Email/Username log in option you will need to have your password ready to complete the conversion process. Click Log In, select the new log in option, and go through the log in process. When you return to Dataverse, look for an option allowing you to convert and enter your password to complete the conversion. The section on Remote Authentication below has more specific information on converting to Institutional Log In, ORCID, GitHub, and Google.

If you need to perform any other conversion (i.e. from Google to GitHub), use the Support link at the top of the page for assistance.

## **Reset Account Password**

Only Dataverse accounts using the Username/Email log in option have an associated password stored (securely!) in Dataverse. If you cannot remember this password, click on Log In in the top right corner of any page and click the “Forgot Password?” link below where you would enter your username/email and password. Enter your email address and click “Submit Password Request” to receive an email with a link to reset your password.

Please note that if you have forgotten your username, you can use this same process to receive your username in an email.

### **1.1.2 Remote Authentication**

Too many passwords? You can set up your Dataverse account to use log in credentials from one of the following remote authentication providers. This way, you can log in using your existing credentials from another service.

#### **Institutional Log In**

Institutional log in allows you to use your log in information for your university (e.g. HarvardKey at Harvard) to log in to your Dataverse account.

#### **Create a Dataverse account using Institutional Log In**

1. Click the “Log In” link in the navbar.
2. Select the “Your Institution” button under the “Other options” header
3. Using the dropdown menu, select your institution then click the Continue button to go to your institution’s log in page.
4. After you put in your institutional credentials successfully, you will be brought back to Dataverse to confirm your account information, and click “Create Account”.
5. A username has been selected for you. You won’t use this username to log in but it will appear next to your name when other users search for you to assign permissions within the system. To see what your username is, click on your name in the top right corner and click Account Information.

If you do not find your institution listed, you will need to request that it is added to the Research & Scholarship category of InCommon. Contact support for assistance on how to get this process started with the identity provider support team at your institution.

### Convert your Dataverse account to use your Institutional Log In

If you already have a Dataverse account associated with the Username/Email log in option, but you want to convert it to use your institutional log in, you can easily do so as long as your account uses an email address from that institution.

1. Go to the Account Information page to confirm that your account email address is the same as your institutional email address. If not, you will need to update your Dataverse account to make them match.
2. Log out of Dataverse.
3. Click the “Log In” link in the navbar.
4. Select the “Your Institution” button under the “Other options” header.
5. Using the dropdown menu, select your institution then click the Continue button to go to your institution’s log in page.
6. After you put in your institutional credentials successfully, you will be brought back to Dataverse to confirm your account information.
7. Enter your current password for your Dataverse account and click “Convert Account”.
8. Now you have finished converting your Dataverse account to use your institutional log in.

### Convert your Dataverse account away from your Institutional Log In

If you are leaving your institution and need to convert your Dataverse account to the Dataverse Username/Email log in option, you will need to contact support for the Dataverse installation you are using. On your account page, there is a link that will open a popup form to contact support for assistance.

### ORCID Log In

You can set up your Dataverse account to allow you to log in using your ORCID credentials. ORCID® is an independent non-profit effort to provide an open registry of unique researcher identifiers and open services to link research activities and organizations to these identifiers. Learn more at [orcid.org](http://orcid.org).

### Create a Dataverse account using ORCID

1. Click the “Log In” link in the navbar.
2. Click the “ORCID” button under the “Other options” header.
3. Click the “Log In with ORCID” button to go to the ORCID website.
4. If you do not already have an ORCID account, you can create one on this page. If you already have an ORCID account, click on “Sign in” and then enter your login under the “Personal account” tab.
5. After you put in your ORCID credentials successfully, you will be brought back to Dataverse to confirm the creation of your Dataverse account. If your ORCID account’s privacy settings permit it, the email address you’ve linked to your ORCID account will be suggested to you as an option. You can use this email if you like, or you can use any other email you might wish. If you have entered employment information within your ORCID account, the name of your employer will be suggested for the “Affiliation” field.

## Convert your Dataverse account to use ORCID for log in

If you already have a Dataverse account associated with the Username/Email log in option, but you want to convert it to use ORCID for log in, follow these steps:

1. Log out of Dataverse.
2. Click the “Log In” link in the navbar.
3. Click the “ORCID” button under the “Other options” header.
4. Click the “Log In with ORCID” button to go to the ORCID website.
5. If you do not already have an ORCID account, you can create one on this page. If you already have an ORCID account, click on “Sign in” and then enter your login under the “Personal account” tab.
6. After you put in your ORCID credentials successfully, you will be brought back to Dataverse. Click the “convert your account” link.
7. Enter your username and password for your Dataverse account and click “Convert Account”.
8. Now you have finished converting your Dataverse account to use ORCID for log in.

## Convert your Dataverse account away from ORCID for log in

If you don’t want to log in to Dataverse using ORCID any more, you will want to convert your Dataverse account to the Dataverse Username/Email log in option. To do this, you will need to contact support for the Dataverse installation you are using. On your account page, there is a link that will open a popup form to contact support for assistance.

## GitHub and Google Log In

You can also convert your Dataverse account to use authentication provided by GitHub or Google. These options may be found in the “Other options” section of the log in page, and function similarly to how ORCID is outlined above. If you would like to convert your account away from using one of these services for log in, then you can follow the same steps as listed above for converting away from the ORCID log in.

### 1.1.3 My Data

The My Data section of your account page displays a listing of all the dataverses, datasets, and files you have either created, uploaded or that you have a role assigned on. If you see unexpected dataverses or datasets in your My Data page, it might be because someone has assigned your account a role on those dataverses or datasets. For example, some institutions automatically assign the “File Downloader” role on their datasets to all accounts using their institutional login.

You are able to filter through all the dataverses, datasets, and files listed on your My Data page using the filter box. You may also use the facets on the left side to only view a specific Publication Status or Role.

### 1.1.4 Notifications

Notifications appear in the notifications tab on your account page and are also displayed as a number next to your account name.

You will receive a notification when:

- You’ve created your account

- You've created a dataverse or added a dataset
- Another Dataverse user has requested access to a restricted file in one of your datasets

Notifications will only be emailed one time even if you haven't read the notification on the Dataverse site.

## 1.1.5 API Token

### What APIs Are and Why They Are Useful

API stands for "Application Programming Interface" and Dataverse APIs allow you to take advantage of integrations with other software that may have been set up by admins of your installation of Dataverse. See the *External Tools* and *Integrations* sections of the Admin Guide for examples of software that is commonly integrated with Dataverse.

Additionally, if you are willing to write a little code (or find someone to write it for you), APIs provide a way to automate parts of your workflow. See the *Getting Started with APIs* section of the API Guide for details.

### How Your API Token Is Like a Password

In many cases, such as when depositing data, an API Token is required to interact with Dataverse APIs. The word "token" indicates a series of letters and numbers such as `c6527048-5bdc-48b0-a1d5-ed1b62c8113b`. Anyone who has your API Token can add and delete data as you so you should treat it with the same care as a password.

### How to Create Your API Token

To create your API token, click on your account name in the navbar, then select "API Token" from the dropdown menu. In this tab, click "Create Token".

### How to Recreate Your API Token

If your API Token becomes compromised or has expired, click on your account name in the navbar, then select "API Token" from the dropdown menu. In this tab, click "Recreate Token".

### Additional Information about API Tokens and Dataverse APIs

Dataverse APIs are documented in the *API Guide* but the following sections may be of particular interest:

- *Getting Started with APIs*
- *API Tokens and Authentication*
- *Frequently Asked Questions*

## 1.2 Finding and Using Data

### Contents:

- *Finding Data*
  - *Basic Search*

- \* *Sorting and Viewing Search Results*
  - *Advanced Search*
  - *Browsing Dataverse*
  - *Saved Search*
- *Using Data*
  - *View Dataverses + Datasets*
  - *View Files*
  - *File Search within Datasets*
  - *Tree View*
  - *Cite Data*
  - *Download Files*
    - \* *Tabular Data*
    - \* *Downloading via URL*
    - \* *Downloading a Dataverse Package via URL*
    - \* *Downloading a Dataverse Package via rsync*
  - *Explore Data*

### 1.2.1 Finding Data

Without logging in to Dataverse, users can browse Dataverse, search for dataverses, datasets, and files, view dataset descriptions and files for published datasets, and subset, analyze, and visualize data for published (restricted & not restricted) data files. To view an unpublished dataverse, dataset, or file, a user will need to be given permission from that dataverse’s administrator to access it.

A user can search within a specific dataverse for the dataverses, datasets, and files it contains by using the search bar and facets displayed on that dataverse’s page.

#### Basic Search

You can search the entire contents of the Dataverse installation, including dataverses, datasets, and files. You can access the search by clicking the “Search” button in the header of every page. The search bar accepts search terms, queries, or exact phrases (in quotations).

#### Sorting and Viewing Search Results

**Facets:** to the left of the search results, there are several facets a user can click on to narrow the number of results displayed.

- Choosing a facet: to choose a facet to narrow your results by, click on that facet.
- Removing a facet: A chosen facet can be removed by clicking on the X on it, either in the facets panel to the left, or above the results.

- Viewing more or fewer facets: Each category in the facets panel lists the top 5 most common facets from that category. To view more, click on “More...” in the bottom right of that category. Once you’ve chosen to see more, an option to view less will appear in the bottom left of the facet.

**Result cards: after entering a search term or query, result cards that match your term or query appear underneath the search bar**

- Relevancy of results: each result card shows which metadata fields match the search query or term you entered into the search bar, with the matching term or query bolded. If the search term or query was found in the title or name of the dataverse, dataset, or file, the search term or query will be bolded within it.

**Other basic search features:**

- Sorting results: search results can be sorted by name (A-Z or Z-A), by date (newest or oldest), or by relevancy of results. The sort button can be found above the search results, in the top right.
- Bookmarkable URLs: search URLs can be copied and sent to a fellow researcher, or can be bookmarked for future sessions.

### Advanced Search

To perform an advanced search, click the “Advanced Search” link next to the search bar. There you will have the ability to enter search terms for dataverses, dataset metadata (citation and domain-specific), and file-level metadata. If you are searching for tabular data files you can also search at the variable level for name and label. To find out more about what each field searches, hover over the field name for a detailed description of the field.

### Browsing Dataverse

In Dataverse, browsing is the default view when a user hasn’t begun a search on the root dataverse page or on a specific dataverse’s page. When browsing, only dataverses and datasets appear in the results list and the results can be sorted by Name (A-Z or Z-A) and by Newest or Oldest. You can toggle the “Files” facet on the left to include files in the results list.

### Saved Search

Saved Search is currently an experimental feature only available to superusers. Please see the *Native API* section of the API Guide for more information.

## 1.2.2 Using Data

### View Dataverses + Datasets

After performing a search and finding the dataverse or dataset you are looking for, click on the name of the dataverse or dataset or on the thumbnail image to be taken to the page for that dataverse or dataset. Once on a dataverse page, you can view the dataverses, datasets, and files within that dataverse.

Once on a dataset page, you will see the title, citation, description, and several other fields, as well as a button to email the dataset contact and a button to share the dataset on social media. Below that information, the files, metadata, terms of use, and version information for the dataset are available.

## View Files

Files in Dataverse each have their own page that can be reached through the search results or through the Files table on their parent dataset's page. The dataset page and file page offer much the same functionality in terms of viewing and editing files, with a few small exceptions. The file page includes the file's persistent identifier (DOI or handle), which can be found under the Metadata tab. Also, the file page's Versions tab gives you a version history that is more focused on the individual file rather than the dataset as a whole.

## File Search within Datasets

Datasets containing multiple files offer a file search function. On the Dataset page, under the Files tab, you'll see a search bar you can use to locate an individual file. It searches within the filename and file description. Performing a search will filter the file table to list only files matching your search. After you perform a search, if you'd like to return to the full list of files, just perform an empty search.

Under the search bar, you'll see file search facets you can use to filter the dataset's files by file type, access level, and file tags (see the example below).

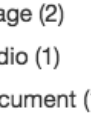
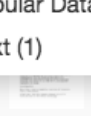




tree\*

Find

Filter by  
 File Type: All ▾ Access: All ▾ File Tag: All ▾

All

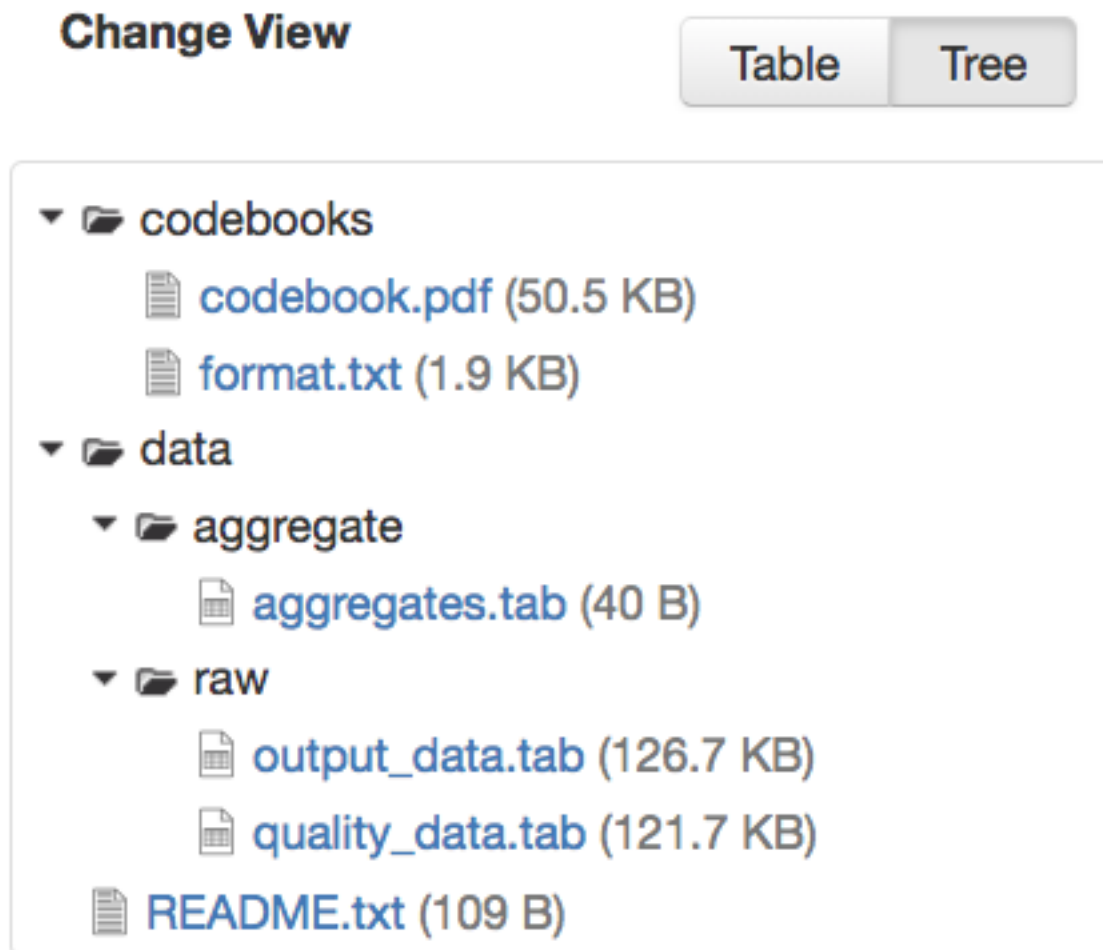
- Image (2)
- Audio (1)
- Document (1)
- Tabular Data (1)
- Text (1)

	<b>g</b> Image - 26.1 KB - May 14, 2019 - 0 Downloads SHA-1: ce332f38cabad13fc74f0619a4437e9ecd260204 <b>Documentation</b>
	<b>ok.pdf</b> PDF - 50.5 KB - May 14, 2019 - 0 Downloads SHA-1: 32ab0d27cc5444885dd817a2156fa2523658abd3 <b>Documentation</b>
<input type="checkbox"/>	 <b>treeinfo.txt</b> Plain Text - 56 B - May 14, 2019 - 0 Downloads SHA-1: 32e66f4013cd1290d4323e1d234b7ea51d0ee25a <b>Documentation</b>
<input type="checkbox"/>	 <b>trees.jpg</b> JPEG Image - 106.7 KB - May 14, 2019 - 0 Downloads SHA-1: aff9ed2e3f888ce1c3738b1b2dda2888a762d230 <b>Illustration</b>
<input type="checkbox"/>	 <b>treesound.mp3</b> audio/mpeg - 81 B - May 14, 2019 - 0 Downloads SHA-1: d83ae761d796991b6d50faf79fbfcdfb0a9e6b2a <b>Soundtrack</b>
<input type="checkbox"/>	 <b>treestata.tab</b> Tabular Data - 4 B - May 14, 2019 - 0 Downloads 1 Variables, 1 Observations - UNF:6:kA2n0P2XnO2MxvaXKfEUAg== <b>Data</b>

(To provide these search facets, we rely on the Solr search engine. Only the latest published version and any draft version of each dataset are indexed in Solr. Because of that, facets cannot be offered for older versions of a dataset.)

### Tree View

Files can be organized in one or more folders (directories) within a dataset. If the folder structure is defined, the Dataset Page will present an option for switching between the traditional table view, and the tree-like view showing folder and file hierarchy, as in the example below:



### Cite Data

You can find the citation for the dataset at the top of the dataset page in a blue box. Additionally, there is a Cite Data button that offers the option to download the citation as EndNote XML, RIS Format, or BibTeX Format.

### Download Files

Within the Files tab on a dataset page, you can download the files in that dataset. To download more than one file at a time, select the files you would like to download and then click the Download button above the files. The selected files will download in .zip format that preserves any folder structure that the dataset owner had set up.



You may also download a file from its file page by clicking the Download button in the upper right corner of the page, or by *Downloading via URL* under the Metadata tab on the lower half of the page.

Tabular data files offer additional options: You can explore using any data exploration or visualization *External Tools* (if they have been enabled) by clicking the Explore button, or choose from a number of tabular-data-specific download options available as a dropdown under the Download button.

## Tabular Data

Ingested files can be downloaded in several different ways.

- The default option is to download a tab-separated-value file which is an easy and free standard to use.
- The original file, which may be in a proprietary format which requires special software
- Rdata format if the installation has configured this
- The variable metadata for the file in DDI format
- A subset of the columns of the data

## Downloading via URL

Dataverse displays a plaintext URL for the location of the file under the Metadata tab on the dataset page. This Download URL can be used to directly access the file via API (or in a web browser, if needed). When downloading larger files, in order to ensure a reliable, resumable download, we recommend using [GNU Wget](#) in a command line terminal or using a download manager software of your choice.

Certain files do not provide Download URLs for technical reasons: those that are restricted, have terms of use associated with them, or are part of a dataverse with a guestbook enabled.

## Downloading a Dataverse Package via URL

Dataverse Packages are typically used to represent extremely large files or bundles containing a large number of files. Dataverse Packages are often too large to be reliably downloaded using a web browser. When you click to download a Dataverse Package, instead of automatically initiating the download in your web browser, Dataverse displays a plaintext URL for the location of the file. To ensure a reliable, resumable download, we recommend using [GNU Wget](#) in a command line terminal or using a download manager software of your choice. If you try to simply paste the URL into your web browser then the download may overwhelm your browser, resulting in an interrupted, timed out, or otherwise failed download.

## Downloading a Dataverse Package via rsync

rsync is typically used for synchronizing files and directories between two different systems. Some Dataverse installations allow downloads using rsync, to facilitate large file transfers in a reliable and secure manner.

rsync-enabled Dataverse installations offer a new file download process that differs from traditional browser-based downloading. Instead of multiple files, each dataset uploaded via rsync contains a single “Dataverse Package”. When you download this package you will receive a folder that contains all files from the dataset, arranged in the exact folder structure in which they were originally uploaded.

In a dataset containing a Dataverse Package, the information to download and/or access is in two places. You can find it on the **dataset page** under the **Files** tab, and on the **file page** under the **Data Access** tab. If the data is locally available to you (on a shared drive, for example) you will find the folder path to access the data locally. To download,

use one of the rsync commands provided. There may be multiple commands, each corresponding to a different mirror that hosts the Dataverse Package. Go outside your browser and open a terminal (AKA command line) window on your computer. Use the terminal to run the command that corresponds with the mirror of your choice. It's usually best to choose the mirror that is geographically closest to you. Running this command will initiate the download process.

After you've downloaded the Dataverse Package, you may want to double-check that your download went perfectly. Under **Verify Data**, you'll find a command that you can run in your terminal that will initiate a checksum to ensure that the data you downloaded matches the data in Dataverse precisely. This way, you can ensure the integrity of the data you're working with.

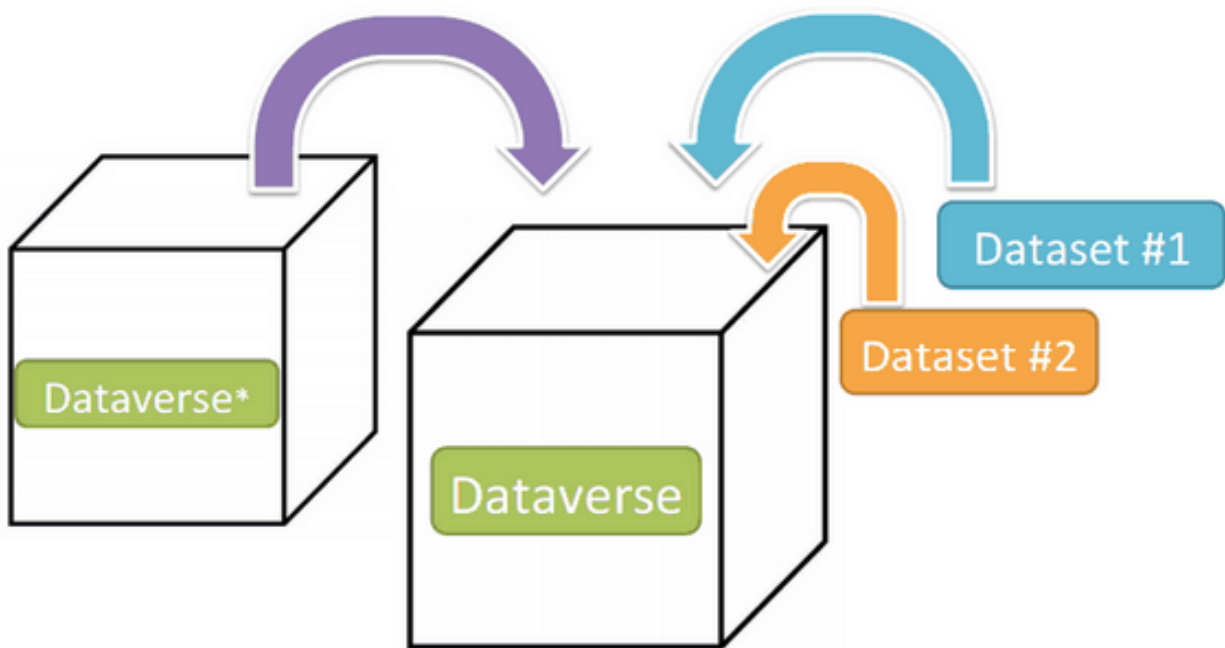
## Explore Data

Please see the *Data Exploration Guide* to get started.

## 1.3 Dataverse Management

A dataverse is a container for datasets (research data, code, documentation, and metadata) and other dataverses, which can be setup for individual researchers, departments, journals and organizations.

### Schematic Diagram of a Dataverse in Dataverse 4.0



### Container for your **Datasets** and/or **Dataverses\***

\* Dataverses can now contain other Dataverses (this replaces Collections & Subnetworks)

Once a user creates a dataverse they, by default, become the administrator of that dataverse. The dataverse administrator has access to manage the settings described in this guide.

**Contents:**

- *Create a Dataverse (Within the “Root” Dataverse)*
- *Edit Dataverse*
  - *General Information*
  - *Theme*
  - *Widgets*
    - \* *Dataverse Search Box Widget*
    - \* *Dataverse Listing Widget*
    - \* *Adding Widgets to an OpenScholar Website*
  - *Roles & Permissions*
    - \* *Setting Access Configurations*
    - \* *Assigning Roles to Users and Groups*
  - *Dataset Templates*
  - *Dataset Guestbooks*
  - *Featured Dataverses*
- *Dataset Linking*
- *Dataverse Linking*
- *Publish Your Dataverse*

**1.3.1 Create a Dataverse (Within the “Root” Dataverse)**

Creating a dataverse is easy but first you must be a registered user (see *Account Creation + Management*).

1. Once you are logged in click on the “Add Data” button and in the dropdown menu select “New Dataverse”.

2. **Once on the “New Dataverse” page fill in the following fields:**

- **Name:** Enter the name of your dataverse.
- **Identifier:** This is an abbreviation, usually lower-case, that becomes part of the URL for the new dataverse. Special characters (~, ', !, @, #, \$, %, ^, &, and \*) and spaces are not allowed. **Note:** if you change this field in the future, the URL for your Dataverse will change (http://.../'identifier'), which will break older links to the page.
- **Email:** This is the email address that will be used as the contact for this particular dataverse. You can have more than one contact email address for your dataverse.
- **Affiliation:** Add any Affiliation that can be associated with this particular dataverse (e.g., project name, institute name, department name, journal name, etc). This is automatically filled out if you have added an affiliation for your user account.
- **Description:** Provide a description of this dataverse. This will display on the landing page of your dataverse and in the search result list. The description field supports certain HTML tags, if you'd like to format your text (<a>, <b>, <blockquote>, <br>, <code>, <del>, <dd>, <dl>, <dt>, <em>, <hr>, <h1>-<h3>, <i>, <img>, <kbd>, <li>, <ol>, <p>, <pre>, <s>, <sup>, <sub>, <strong>, <strike>, <ul>).

- **Category:** Select a category that best describes the type of dataverse this will be. For example, if this is a dataverse for an individual researcher’s datasets, select *Researcher*. If this is a dataverse for an institution, select *Organization or Institution*.
  - **Choose the sets of Metadata Fields for datasets in this dataverse:** By default the metadata elements will be from the host dataverse that this new dataverse is created in. Dataverse offers metadata standards for multiple domains. To learn more about the metadata standards in Dataverse please check out the *Appendix*.
  - **Select facets for this dataverse:** Choose which metadata fields will be used as facets on your dataverse. These facets will allow users browsing or searching your dataverse to filter its contents according to the fields you’ve selected. For example, if you select “Subject” as a facet, users will be able to filter your dataverse’s contents by subject area. By default, the facets that will appear on your dataverse landing page will be from the host dataverse that this new dataverse was created in, but you can add or remove facets from this default.
3. Selected metadata elements are also used to pick which metadata fields you would like to use for creating templates for your datasets. Metadata fields can be hidden, or selected as required or optional. Once you have selected all the fields you would like to use, you can create your template(s) after you finish creating your dataverse.
  4. Click “Create Dataverse” button and you’re done!

\*Required fields are denoted by a red asterisk.

### 1.3.2 Edit Dataverse

To edit your dataverse, navigate to your dataverse’s landing page and select the “Edit Dataverse” button, where you will be presented with the following editing options:

- *General Information:* edit name, identifier, category, contact email, affiliation, description, Metadata Elements, and facets for your dataverse
- *Theme:* upload a logo for your dataverse, add a link to your department or personal website, add a custom footer image, and select colors for your dataverse in order to brand it
- *Widgets:* get code to add to your website to have your dataverse display on it
- *Permissions:* give Dataverse users permissions to your dataverse, i.e.-can edit datasets, and see which users already have which permissions for your dataverse
- *Dataset Templates:* these are useful when you have several datasets that have the same information in multiple metadata fields that you would prefer not to have to keep manually typing in
- *Dataset Guestbooks:* allows you to collect data about who is downloading the files from your datasets
- *Featured Dataverses:* if you have one or more dataverses, you can use this option to show them at the top of your dataverse page to help others easily find interesting or important dataverses
- **Delete Dataverse:** you are able to delete your dataverse as long as it is not published and does not have any draft datasets

### General Information

The General Information page is how you edit the information you filled in while creating your dataverse. If you need to change or add a contact email address, this is the place to do it. Additionally, you can update the metadata elements used for datasets within the dataverse, change which metadata fields are hidden, required, or optional, and update the facets you would like displayed for browsing the dataverse. If you plan on using templates, you need to select the metadata fields on the General Information page.

Tip: The metadata fields you select as required will appear on the Create Dataset form when someone goes to add a dataset to the dataverse.

## Theme

The Theme features provides you with a way to customize the look of your dataverse. You can:

- Inherit the theme from the parent dataverse. This option is helpful if you'd like consistency across several dataverses that all share the same parent.
- Add or update a logo image, which will appear at the top of your dataverse.
- Add or update a footer image, which will appear at at the bottom of your dataverse.
- Change the colors of the background, links, and text within the header of your dataverse.
- Add or update the tagline for your dataverse, which can provide more information about your organization, journal, institution, etc.
- Add a URL for a website that will be accessed when visitors click the tagline text.

Supported image types for logo images and footer images are JPEG, TIFF, or PNG and should be no larger than 500 KB. The maximum display size for an image file in a dataverse's theme is 940 pixels wide by 120 pixels high.

## Widgets

The Widgets feature provides you with code for you to put on your personal website to have your dataverse displayed there. There are two types of Widgets for a dataverse, a Dataverse Search Box widget and a Dataverse Listing widget. From the Widgets tab on the Theme + Widgets page, you can copy and paste the code snippets for the widget you would like to add to your website. If you need to adjust the height of the widget on your website, you may do so by editing the `heightPx=500` parameter in the code snippet.

### Dataverse Search Box Widget

The Dataverse Search Box Widget will add a search box to your website that is linked to your dataverse. Users are directed to your dataverse in a new browser window, to display the results for search terms entered in the input field.

### Dataverse Listing Widget

The Dataverse Listing Widget provides a listing of all your dataverses and datasets for users to browse, sort, filter and search. When someone clicks on a dataverse or dataset in the widget, it displays the content in the widget on your website. They can download data files directly from the datasets within the widget. If a file is restricted, they will be directed to your dataverse to log in, instead of logging in through the widget on your website.

### Adding Widgets to an OpenScholar Website

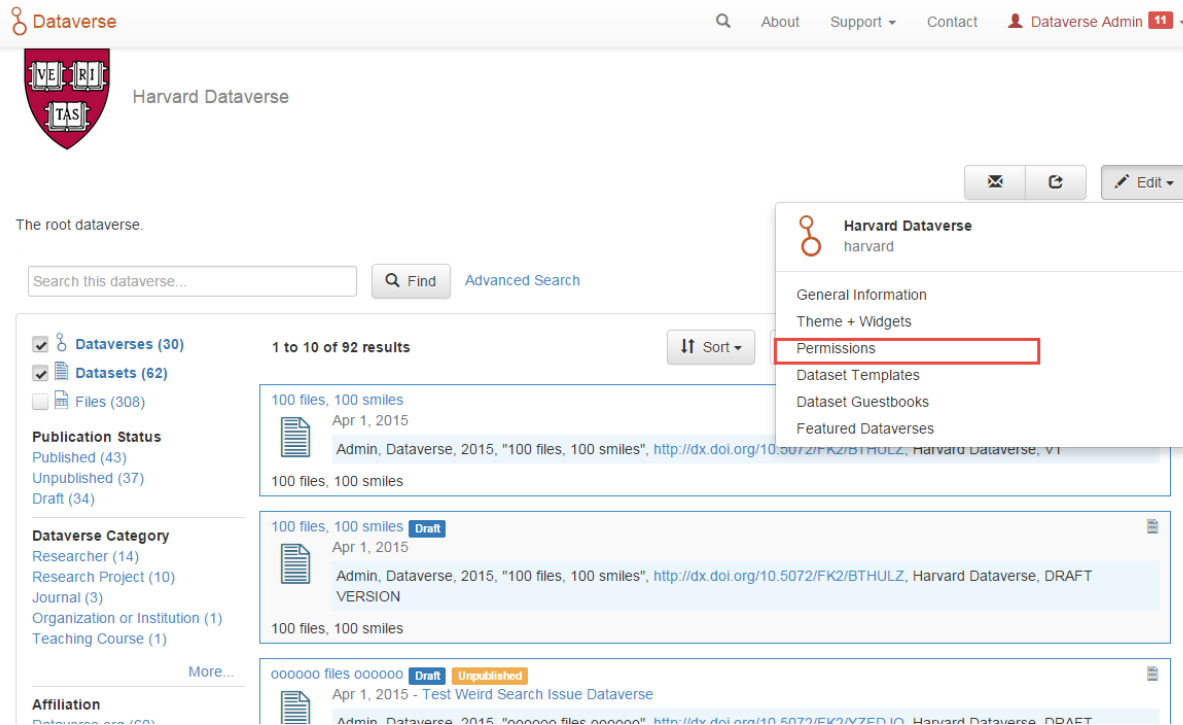
1. Log in to your OpenScholar website
2. Either build a new page or navigate to the page you would like to use to show the Dataverse widgets.
3. Click on the Settings Cog and select Layout
4. At the top right, select Add New Widget and under Misc. you will see the Dataverse Search Box and the Dataverse Listing Widgets. Click on the widget you would like to add, fill out the form, and then drag it to where you would like it to display in the page.

## Roles & Permissions

Dataverse user accounts can be granted roles that define which actions they are allowed to take on specific dataverses, datasets, and/or files. Each role comes with a set of permissions, which define the specific actions that users may take.

Roles and permissions may also be granted to groups. Groups can be defined as a collection of Dataverse user accounts, a collection of IP addresses (e.g. all users of a library's computers), or a collection of all users who log in using a particular institutional login (e.g. everyone who logs in with a particular university's account credentials).

Admins of a dataverse can assign roles and permissions to the users of that dataverse. If you are an admin on a dataverse, then you will find the link to the Permissions page under the Edit dropdown on the dataverse page.



Clicking on Permissions will bring you to this page:

**Permissions** ^

Here is the current access configuration to your dataverse. Edit Access

**Who can add to this dataverse?**  
Anyone adding to this dataverse needs to be given access

**What should be the default role for someone adding datasets to this dataverse?**  
Contributor - Edit metadata, upload files, and edit files, edit Terms, Guestbook, Submit datasets for review

**Users/Groups** ^

Here are all the users and groups that have access to your dataverse. Create Group Assign Roles to Users/Groups

User/Group Name (Affiliation)	ID	Role	Action
Dataverse Admin (Dataverse.org)	@admin	Admin	<span>Remove Assigned Role</span>

**Roles** v

When you access a dataverse’s permissions page, you will see three sections:

**Permissions:** Here you can decide the requirements that determine which types of users can add datasets and sub dataverses to your dataverse, and what permissions they’ll be granted when they do so.

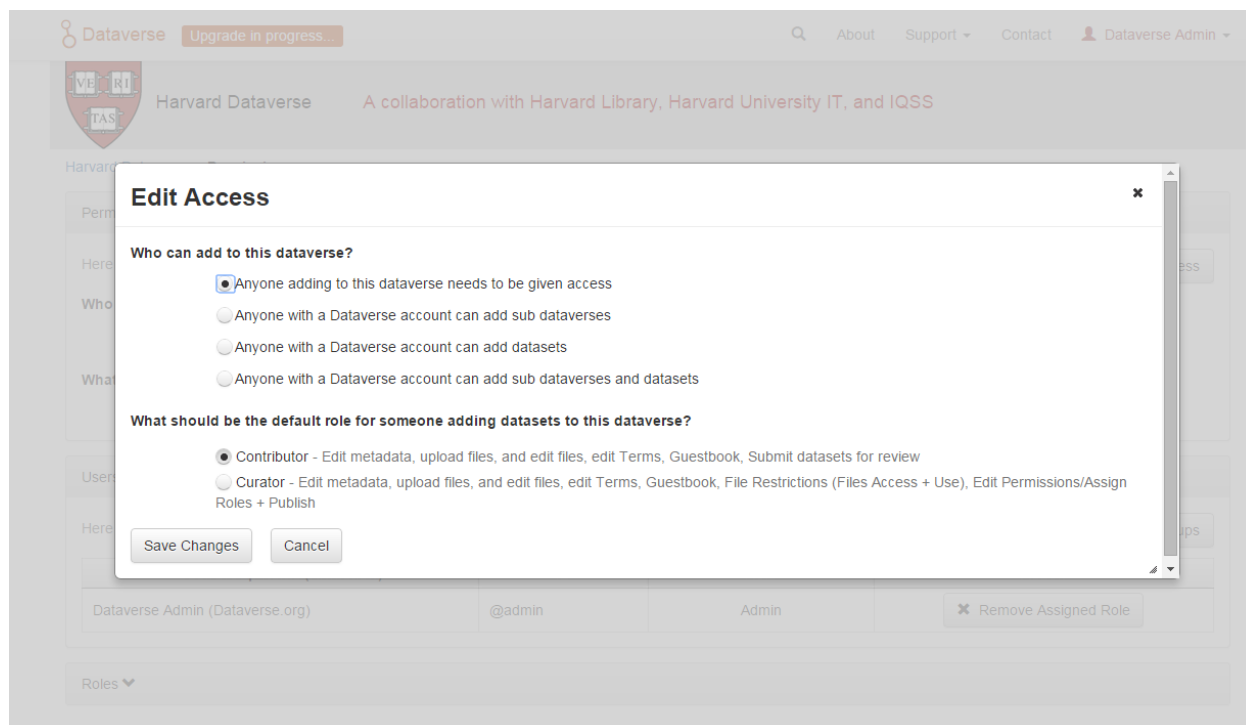
**Users/Groups:** Here you can assign roles to specific users or groups, determining which actions they are permitted to take on your dataverse. You can also reference a list of all users who have roles assigned to them for your dataverse and remove their roles if you please.

**Roles:** Here you can reference a full list of roles that can be assigned to users of your dataverse. Each role lists the permissions that it offers.

Please note that even on a newly created dataverse, you may see user and groups have already been granted role(s) if your installation has `:InheritParentRoleAssignments` set. For more on this setting, see the [Configuration](#) section of the Installation Guide.

## Setting Access Configurations

Under the Permissions tab, you can click the “Edit Access” button to open a box where you can add to your dataverse and what permissions are granted to those who add to your dataverse.



The first question on this page allows you to determine how open your dataverse is to new additions - you can set whether or not the entire userbase (all logged in users) has the ability to add datasets or sub dataverses to your dataverse.

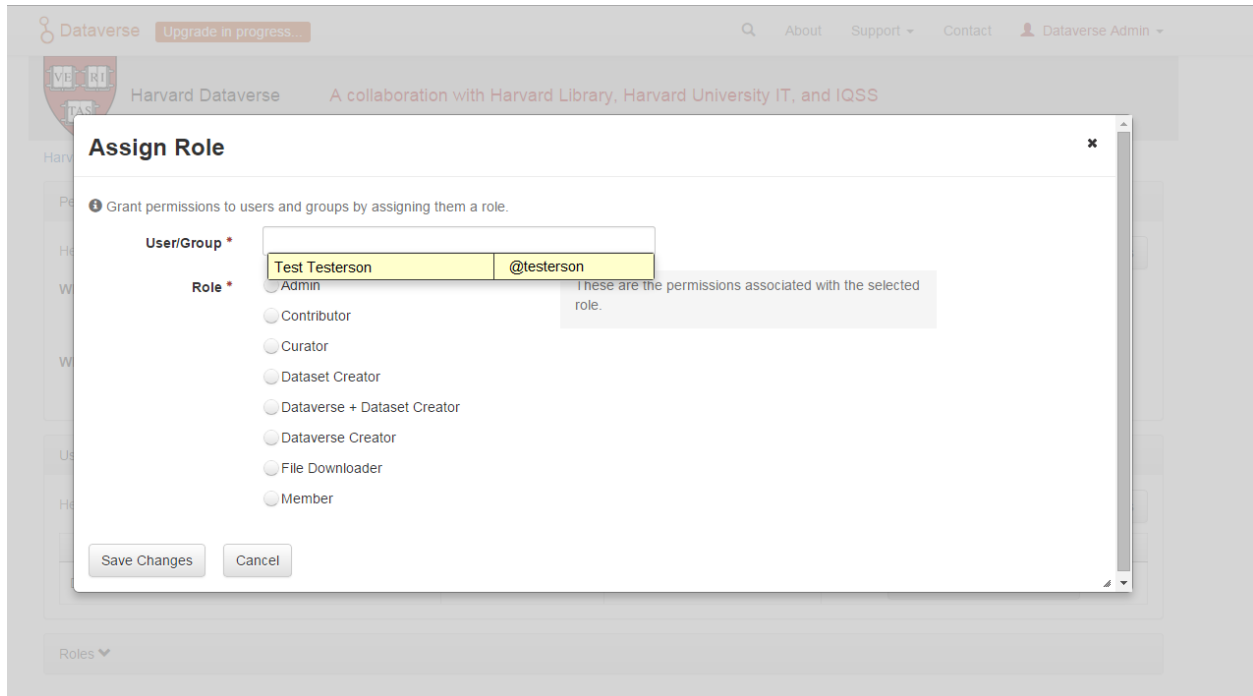
The second question on this page allows you to choose the role (and thus the permissions) granted to users who add a dataset to your dataverse. The role you select will be automatically granted to any user who creates a dataset on your dataverse, on that dataset, at the moment that he or she creates it. The role the user is given determines his or her permissions for the dataset they've created. The key difference between the two roles is that curators can publish their own datasets, while contributors must submit the dataset to be reviewed before publication. Additionally, curators can manage dataset permissions. Note that this setting does not retroactively apply roles to users who have previously added datasets to your dataverse; it only applies to users adding new datasets going forward.

Both of these settings can be changed at any time.

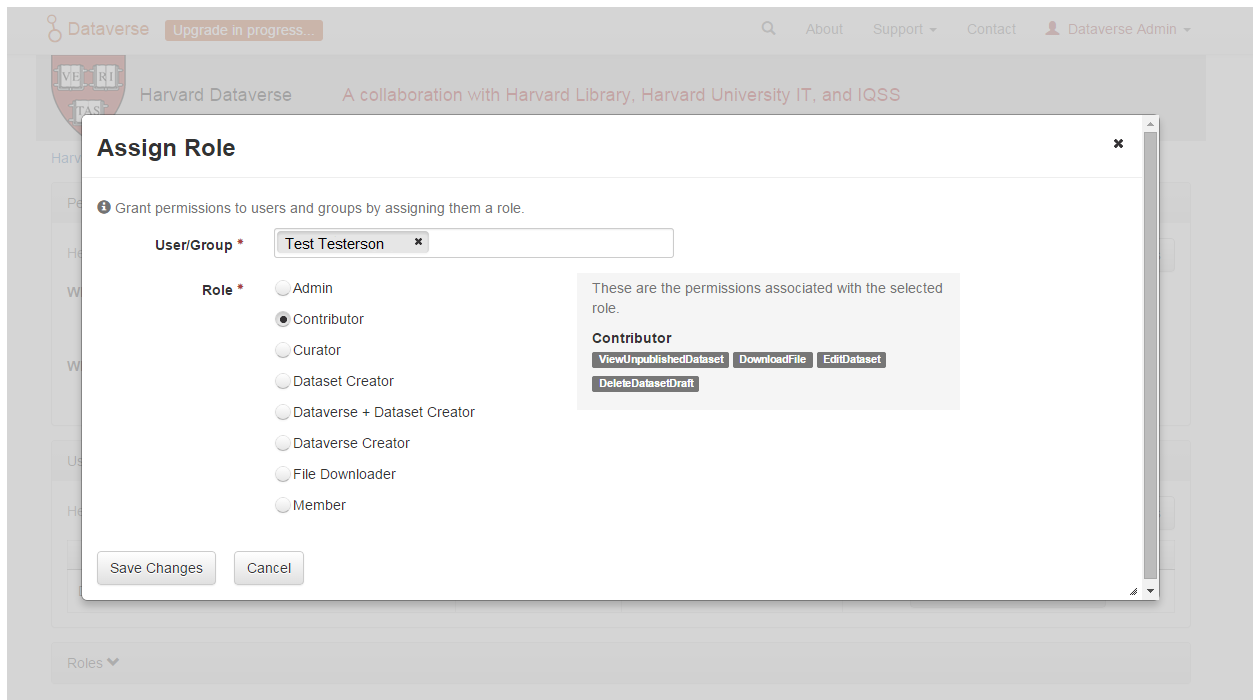
## Assigning Roles to Users and Groups

Under the Users/Groups tab, you can add, edit, or remove the roles granted to users and groups on your dataverse. A role is a set of permissions granted to a user or group when they're using your dataverse. For example, giving your research assistant the "Contributor" role would give her the following self-explanatory permissions on your dataverse and all datasets within your dataverse: "ViewUnpublishedDataset", "DownloadFile", "EditDataset", and "DeleteDatasetDraft". She would, however, lack the "PublishDataset" permission, and thus would be unable to publish datasets on your dataverse. If you wanted to give her that permission, you would give her a role with that permission, like the Curator role. Users and groups can hold multiple roles at the same time if needed. Roles can be removed at any time. All roles and their associated permissions are listed under the "Roles" tab of the same page.





Note that the Dataset Creator role and Contributor role are sometimes confused. The Dataset Creator role is assigned at the dataverse level and allows a user to create new datasets in that dataverse. The Contributor role can be assigned at the dataset level, granting a user the ability to edit *that specific* dataset. Alternatively, the Contributor role can be assigned at the dataverse level, granting the user the ability to edit *all* datasets in that dataverse.



Note: If you need to assign a role to ALL Dataverse user accounts, you can assign the role to the `”:authenticated-users”` group.

### Dataset Templates

Templates are useful when you have several datasets that have the same information in multiple metadata fields that you would prefer not to have to keep manually typing in, or if you want to use a custom set of Terms of Use and Access for multiple datasets in a dataverse. In Dataverse 4.0, templates are created at the dataverse level, can be deleted (so it does not show for future datasets), set to default (not required), or can be copied so you do not have to start over when creating a new template with similar metadata from another template. When a template is deleted, it does not impact the datasets that have used the template already.

How do you create a template?

1. Navigate to your dataverse, click on the Edit Dataverse button and select Dataset Templates.
2. Once you have clicked on Dataset Templates, you will be brought to the Dataset Templates page. On this page, you can 1) decide to use the dataset templates from your parent dataverse 2) create a new dataset template or 3) do both.
3. Click on the Create Dataset Template to get started. You will see that the template is the same as the create dataset page with an additional field at the top of the page to add a name for the template.
4. After adding information into the metadata fields you have information for and clicking Save and Add Terms, you will be brought to the page where you can add custom Terms of Use and Access. If you do not need custom Terms of Use and Access, click the Save Dataset Template, and only the metadata fields will be saved.
5. After clicking Save Dataset Template, you will be brought back to the Manage Dataset Templates page and should see your template listed there now with the make default, edit, view, or delete options.
6. A dataverse does not have to have a default template and users can select which template they would like to use while on the Create Dataset page.
7. You can also click on the View button on the Manage Dataset Templates page to see what metadata fields have information filled in.

\* Please note that the ability to choose which metadata fields are hidden, required, or optional is done on the General Information page for the dataverse.

### Dataset Guestbooks

Guestbooks allow you to collect data about who is downloading the files from your datasets. You can decide to collect account information (username, given name & last name, affiliation, etc.) as well as create custom questions (e.g., What do you plan to use this data for?). You are also able to download the data collected from the enabled guestbooks as Excel files to store and use outside of Dataverse.

How do you create a guestbook?

1. After creating a dataverse, click on the Edit Dataverse button and select Dataset Guestbook
2. By default, guestbooks created in the dataverse your dataverse is in, will appear. If you do not want to use or see those guestbooks, uncheck the checkbox that says Include Guestbooks from Root Dataverse.
3. To create a new guestbook, click the Create Dataset Guestbook button on the right side of the page.
4. Name the guestbook, determine the account information that you would like to be required (all account information fields show when someone downloads a file), and then add Custom Questions (can be required or not required).
5. Hit the Create Dataset Guestbook button once you have finished.

What can you do with a guestbook? After creating a guestbook, you will notice there are several options for a guestbook and appear in the list of guestbooks.

- If you want to use a guestbook you have created, you will first need to click the button in the Action column that says Enable. Once a guestbook has been enabled, you can go to the License + Terms for a dataset and select a guestbook for it.
- There are also options to view, copy, edit, or delete a guestbook.
- Once someone has downloaded a file in a dataset where a guestbook has been assigned, an option to download collected data will appear.

## Featured Dataverses

Featured Dataverses is a way to display sub dataverses in your dataverse that you want to feature for people to easily see when they visit your dataverse.

Click on Featured Dataverses and a pop up will appear. Select which sub dataverses you would like to have appear.

Note: Featured Dataverses can only be used with published dataverses.

### 1.3.3 Dataset Linking

Dataset linking allows a dataverse owner to “link” their dataverse to a dataset that exists outside of that dataverse, so it appears in the dataverse’s list of contents without actually *being* in that dataverse. You can link other users’ datasets to your dataverse, but that does not transfer editing or other special permissions to you. The linked dataset will still be under the original user’s control.

For example, researchers working on a collaborative study across institutions can each link their own individual institutional dataverses to the one collaborative dataset, making it easier for interested parties from each institution to find the study.

In order to link a dataset, you will need your account to have the “Add Dataset” permission on the Dataverse that is doing the linking. If you created the dataverse then you should have this permission already, but if not then you will need to ask the admin of that dataverse to assign that permission to your account. You do not need any special permissions on the dataset being linked.

To link a dataset to your dataverse, you must navigate to that dataset and click the white “Link” button in the upper-right corner of the dataset page. This will open up a window where you can type in the name of the dataverse that you would like to link the dataset to. Select your dataverse and click the save button. This will establish the link, and the dataset will now appear under your dataverse.

There is currently no way to remove established links in the UI. If you need to remove a link between a dataverse and a dataset, please contact the support team for the Dataverse installation you are using.

### 1.3.4 Dataverse Linking

Similarly to dataset linking, dataverse linking allows a dataverse owner to “link” their dataverse to another dataverse, so the dataverse being linked will appear in the linking dataverse’s list of contents without actually *being* in that dataverse. Currently, the ability to link a dataverse to another dataverse is a superuser only feature.

If you need to have a dataverse linked to your dataverse, please contact the support team for the Dataverse installation you are using.

### 1.3.5 Publish Your Dataverse

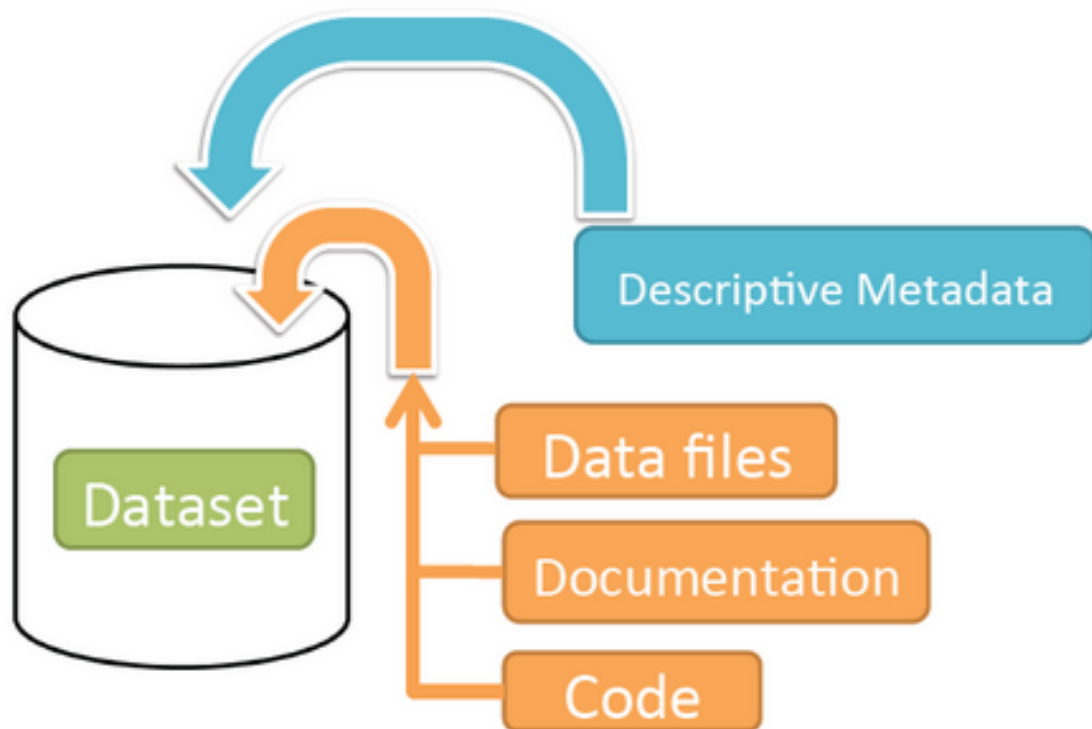
Once your dataverse is ready to go public, go to your dataverse page, click on the “Publish” button on the right hand side of the page. A pop-up will appear to confirm that you are ready to actually Publish, since once a dataverse is made

public, it can no longer be unpublished.

## 1.4 Dataset + File Management

A dataset in Dataverse is a container for your data, documentation, code, and the metadata describing this Dataset.

### Schematic Diagram of a **Dataset** in Dataverse 4.0



Container for your data, documentation, and code.

#### Contents:

- *Supported Metadata*
  - *Supported Metadata Export Formats*
- *Adding a New Dataset*
  - *Supported HTML Fields*
- *File Upload*
  - *HTTP Upload*
  - *Dropbox Upload*

- *rsync + SSH Upload*
    - \* *File Upload Script*
  - *Command-line DVUploader*
    - \* *Usage*
- *File Handling*
  - *Tabular Data Files*
  - *Geospatial*
  - *Astronomy (FITS)*
  - *Compressed Files*
  - *Other File Types*
- *Edit Files*
  - *Edit File Metadata*
  - *Edit File Variable Metadata*
  - *File Path*
  - *File Tags*
- *Replace Files*
- *Terms*
  - *CC0 Public Domain Dedication*
  - *Custom Terms of Use for Datasets*
  - *Restricted Files + Terms of Access*
  - *Guestbook*
- *Roles & Permissions*
  - *Dataset-Level*
  - *File-Level*
- *Data Provenance*
- *Thumbnails + Widgets*
  - *Thumbnails*
  - *Widgets*
    - \* *Dataset Widget*
    - \* *Dataset Citation Widget*
    - \* *Adding Widgets to an OpenScholar Website*
- *Publish Dataset*
- *Submit for Review*
- *Private URL to Review Unpublished Dataset*
- *Dataset Versions*

- *Version Details*
- *Dataset Metrics and Make Data Count*
- *Cloud Storage + Computing*
  - *Cloud Computing*
  - *Cloud Storage Access*
- *Dataset Deaccession*

### 1.4.1 Supported Metadata

A dataset contains three levels of metadata:

1. **Citation Metadata:** any metadata that would be needed for generating a data citation and other general metadata that could be applied to any dataset;
2. **Domain Specific Metadata:** with specific support currently for Social Science, Life Science, Geospatial, and Astronomy datasets; and
3. **File-level Metadata:** varies depending on the type of data file - for more details see *File Handling* section below).

For more details about what Citation and Domain Specific Metadata is supported please see our *Appendix*.

### Supported Metadata Export Formats

Once a dataset has been published, its metadata can be exported in a variety of other metadata standards and formats, which help make datasets more discoverable and usable in other systems, such as other data repositories. On each dataset page’s metadata tab, the following exports are available:

- Dublin Core
- DDI (Data Documentation Initiative Codebook 2.5)
- DDI HTML Codebook (A more human-readable, HTML version of the DDI Codebook 2.5 metadata export)
- DataCite 4
- JSON (native Dataverse format)
- OAI\_ORE
- OpenAIRE
- Schema.org JSON-LD

Each of these metadata exports contains the metadata of the most recently published version of the dataset.

### 1.4.2 Adding a New Dataset

1. Navigate to the dataverse in which you want to add a dataset.
2. Click on the “Add Data” button and select “New Dataset” in the dropdown menu.
3. To quickly get started, enter at minimum all the required fields with an asterisk (e.g., the Dataset Title, Author, Description, Contact Email and Subject) to get a Data Citation with a DOI.

4. Scroll down to the “Files” section and click on “Select Files to Add” to add all the relevant files to your Dataset. You can also upload your files directly from your Dropbox. **Tip:** You can drag and drop or select multiple files at a time from your desktop directly into the upload widget. Your files will appear below the “Select Files to Add” button where you can add a description and tags (via the “Edit Tag” button) for each file. Additionally, an MD5 checksum will be added for each file. If you upload a tabular file a *Universal Numerical Fingerprint (UNF)* will be added to this file.
5. Click the “Save Dataset” button when you are done. Your unpublished dataset is now created.

Note: You can add additional metadata once you have completed the initial dataset creation by going to Edit Dataset > Metadata.

## Supported HTML Fields

We currently only support the following HTML tags for any of our textbox metadata fields (i.e., Description) : <a>, <b>, <blockquote>, <br>, <code>, <del>, <dd>, <dl>, <dt>, <em>, <hr>, <h1>-<h3>, <i>, <img>, <kbd>, <li>, <ol>, <p>, <pre>, <s>, <sup>, <sub>, <strong>, <strike>, <ul>.

### 1.4.3 File Upload

The Dataverse software offers multiple methods of uploading files to a dataset. These upload methods are configurable by the administrator of a Dataverse installation, so you might not see some of these options on the Dataverse site you’re using.

If there are multiple upload options available, then you must choose which one to use for your dataset. A dataset may only use one upload method. Once you upload a file using one of the available upload methods, that method is locked in for that dataset. If you need to switch upload methods for a dataset that already contains files, then please contact Support by clicking on the Support link at the top of the application.

You can upload files to a dataset while first creating that dataset. You can also upload files after creating a dataset by clicking the “Edit” button at the top of the dataset page and from the dropdown list selecting “Files (Upload)” or clicking the “Upload Files” button above the files table in the Files tab. From either option you will be brought to the Upload Files page for that dataset.

Certain file types in Dataverse are supported by additional functionality, which can include downloading in different formats, file-level metadata preservation, file-level data citation with UNFs, and exploration through data visualization and analysis. See the *File Handling* section of this page for more information.

## HTTP Upload

HTTP Upload is a common browser-based file upload tool you may be familiar with from other web applications. You can upload files via HTTP by selecting them from your browser or dragging and dropping them into the upload widget.

Once you have uploaded files, you will be able to edit file metadata, restrict access to files<sup>1</sup>, and/or add tags. Click “Save Changes” to complete the upload. If you uploaded a file by mistake, you can delete it before saving by clicking the checkbox to select the file, and then clicking the “Delete” button above the Files Table.

File upload limit size varies based on Dataverse installation. The file upload size limit can be found in the text above the HTTP upload widget. If you need to upload a very large file or a very large *number* of files, consider using rsync + SSH upload if your installation of Dataverse offers it.

<sup>1</sup> Some Dataverse installations do not allow this feature.

### Dropbox Upload

Some Dataverse installations support the ability to upload files directly from Dropbox. To do so, click the “Upload from Dropbox” button, log in to Dropbox in the pop-up window, and select the files you’d like to transfer over.

### rsync + SSH Upload

rsync is typically used for synchronizing files and directories between two different systems, using SSH to connect rather than HTTP. Some Dataverse installations allow uploads using rsync, to facilitate large file transfers in a reliable and secure manner.

### File Upload Script

An rsync-enabled Dataverse installation has a file upload process that differs from the traditional browser-based upload process you may be used to. In order to transfer your data to Dataverse’s storage, you will need to complete the following steps:

1. Create your dataset. In rsync-enabled Dataverse installations, you cannot upload files until the dataset creation process is complete. After you hit “Save Dataset” on the Dataset Creation page, you will be taken to the page for your dataset.
2. On the dataset page, click the “+ Upload Files” button. This will open a box with instructions and a link to the file upload script.
3. Make sure your files are ready for upload. You will need to have one directory that you can point the upload script to. All files in this directory and in any subdirectories will be uploaded. The directory structure will be preserved, and will be reproduced when your dataset is downloaded from Dataverse. Note that your data will be uploaded in the form of a data package, and each dataset can only host one such package. Be sure that all files you want to include are present before you upload.
4. Download the rsync file upload script by clicking the “Download Script” button in the Upload Files instruction box. There are no requirements for where you save the script; put it somewhere you can find it. Downloading the upload script will put a temporary lock on your dataset to prepare it for upload. While your dataset is locked, you will not be able to delete or publish your dataset, or edit its metadata. Once you upload your files and Dataverse processes them, your dataset will be automatically unlocked and these disabled functions will be enabled again. If you have downloaded the script and locked your dataset, but you have then changed your mind and decided *not* to upload files, please contact Support about unlocking your dataset.
5. To begin the upload process, you will need to run the script you downloaded. For this, you will have to go outside your browser and open a terminal (AKA command line) window on your computer. Use the terminal to navigate to the directory where you saved the upload script, and run the command that the Upload Files instruction box provides. This will begin the upload script. Please note that this upload script will expire 7 days after you downloaded it. If it expires and you still need to use it, simply download the script from Dataverse again.

**Note:** Unlike other operating systems, Windows does not come with rsync supported by default. We have not optimized this feature for Windows users, but you may be able to get it working if you install the right Unix utilities. (If you have found a way to get this feature working for you on Windows, you can contribute it to our project. Please reference our [Contributing to Dataverse](#) document in the root of the source tree.)

6. Follow the instructions provided by the upload script running in your terminal. It will direct you to enter the full path of the directory where your dataset files are located, and then it will start the upload process. Once you’ve initiated the upload, if you need to cancel it then you can do so by canceling the script running in your terminal window. If your upload gets interrupted, you can resume it from the same point later.



7. Once the upload script completes its job, Dataverse will begin processing your data upload and running a checksum validation. This may take some time depending on the file size of your upload. During processing, you will see a blue bar at the bottom of the dataset page that reads “Upload in progress...”
8. Once processing is complete, you will be notified. At this point you can publish your dataset and your data will be available for download on the dataset page.

**Note:** A dataset can only hold one data package. If you need to replace the data package in your dataset, contact Support.

## Command-line DVUploader

The open-source DVUploader tool is a stand-alone command-line Java application that uses the Dataverse API to upload files to a specified Dataset. Since it can be installed by users, and requires no server-side configuration, it can be used with any Dataverse installation. It is intended as an alternative to uploading files through the Dataverse web interface in situations where the web interface is inconvenient due to the number of files or file locations (spread across multiple directories, mixed with files that have already been uploaded or file types that should be excluded) or the need to automate uploads. Since it uses the Dataverse API, transfers are limited in the same ways as HTTP uploads through the Dataverse web interface in terms of size and performance. The DVUploader logs its activity and can be killed and restarted as desired. If stopped and resumed, it will continue processing from where it left off.

## Usage

The DVUploader is open source and is available as source, as a Java jar, and with documentation at <https://github.com/IQSS/dataverse-uploader>. The DVUploader requires Java 1.8+. Users will need to install Java if they don't already have it and then download the DVUploader-v1.0.0.jar file. Users will need to know the URL of the Dataverse server, the DOI of their existing Dataverse Dataset, and have generated a Dataverse API Key (an option in the user's profile menu).

Basic usage is to run the command:

```
java -jar DVUploader-v1.0.0.jar -server=<Dataverse server URL> -did=<Dataset DOI> -
  ↪key=<User's API Key> <file or directory list>
```

Additional command line arguments are available to make the DVUploader list what it would do without uploading, limit the number of files it uploads, recurse through sub-directories, verify fixity, exclude files with specific extensions or name patterns, and/or wait longer than 60 seconds for any Dataverse ingest lock to clear (e.g. while the previously uploaded file is processed, as discussed in the *File Handling* section below).

DVUploader is a community-developed tool, and its creation was primarily supported by the Texas Digital Library. Further information and support for DVUploader can be sought at [the project's GitHub repository](#).

### 1.4.4 File Handling

Certain file types in Dataverse are supported by additional functionality, which can include downloading in different formats, file-level metadata preservation, file-level data citation; and exploration through data visualization and analysis. See the sections below for information about special functionality for specific file types.

#### Tabular Data Files

Files in certain formats - Stata, SPSS, R, Excel(xlsx) and CSV - may be ingested as tabular data (see *Tabular Data File Ingest* section of the User Guide for details). Tabular data files can be further explored and manipulated with *TwoRavens* - a statistical data exploration application integrated with Dataverse, as well as other *External Tools* if they

have been enabled in the installation of Dataverse you are using. TwoRavens allows the user to run statistical models, view summary statistics, download subsets of variable vectors and more. To start, click on the “Explore” button, found next to each relevant tabular file (the application will be opened in a new window). Create and download your subset using [TwoRavens](#). See the [TwoRavens documentation section](#) for more information.

Additional download options available for tabular data (found in the same drop-down menu under the “Download” button):

- As tab-delimited data (with the variable names in the first row);
- The original file uploaded by the user;
- Saved as R data (if the original file was not in R format);
- Variable Metadata (as a [DDI Codebook XML](#) file);
- Data File Citation (currently in either RIS, EndNote XML, or BibTeX format);
- All of the above, as a zipped bundle.

### Geospatial

Geospatial [shapefiles](#) can be further explored and manipulated through our integration with [WorldMap](#), a geospatial data visualization and analysis tool developed by the [Center for Geographic Analysis](#) at Harvard University. A shapefile is a set of files, often uploaded/transferred in .zip format. This set may contain up to 15 files. A minimum of 3 specific files (.shp, .shx, .dbf) are needed to be a valid shapefile and a 4th file (.prj) is required for WorldMap—or any type of meaningful visualization.

For ingest into Dataverse and connecting to WorldMap, these 4 files are the minimum required:

- .shp - shape format; the feature geometry itself
- .shx - shape index format; a positional index of the feature geometry to allow seeking forwards and backwards quickly
- .dbf - attribute format; columnar attributes for each shape, in dBase IV format
- .prj - projection format; the coordinate system and projection information, a plain text file describing the projection using well-known text format

For a zipped shapefile, we require 4 files with these extensions. Other files may be included within the zipped shapefile, but they are not required:

- .shp
- .shx
- .prj
- .dbf

For example, if these files were included within a .zip, the “Map Data” button would appear:

- subway\_line.shp
- subway\_line.shx
- subway\_line.prj
- subway\_line.dbf

Once you publish your dataset with your shape files, you will be able to use the “Map Data” button using [GeoConnect](#) to visualize and manipulate these files for users to Explore this geospatial data using the [WorldMap](#) interface. Please note: In order to map your data file, a copy will be sent to Harvard’s [WorldMap](#) platform. You have the ability to delete any maps, and associated data, from the Harvard WorldMap platform, at any time.

## Astronomy (FITS)

Metadata found in the header section of [Flexible Image Transport System \(FITS\) files](#) are automatically extracted by Dataverse, aggregated and displayed in the Astronomy Domain-Specific Metadata of the Dataset that the file belongs to. This FITS file metadata, is therefore searchable and browsable (facets) at the Dataset-level.

## Compressed Files

Compressed files in .zip format are unpacked automatically. If a .zip file fails to unpack for whatever reason, it will upload as is. If the number of files inside are more than a set limit (1,000 by default, configurable by the Administrator), you will get an error message and the .zip file will upload as is.

If the uploaded .zip file contains a folder structure, Dataverse will keep track of this structure. A file's location within this folder structure is displayed in the file metadata as the File Path. When you download the contents of the dataset, this folder structure will be preserved and files will appear in their original locations.

These folder names are subject to strict validation rules. Only the following characters are allowed: the alphanumerics, '\_', '-', '.', and ' ' (white space). When a zip archive is uploaded, the folder names are automatically sanitized, with any invalid characters replaced by the '.' character. Any sequences of dots are further replaced with a single dot. For example, the folder name `data&info/code=@137` will be converted to `data.info/code.137`. When uploading through the Web UI, the user can change the values further on the edit form presented, before clicking the 'Save' button.

---

**Note:** If you upload multiple .zip files to one dataset, any subdirectories that are identical across multiple .zips will be merged together when the user downloads the full dataset.

---

## Other File Types

There are several advanced options available for certain file types.

- Image files: .jpg, .png, and .tif files are able to be selected as the default thumbnail for a dataset. The selected thumbnail will appear on the search result card for that dataset.
- SPSS files: SPSS files can be tagged with the language they were originally coded in. This is found by clicking on Advanced Options and selecting the language from the list provided.

## 1.4.5 Edit Files

### Edit File Metadata

Go to the dataset you would like to edit, where you will see the listing of files. Select the files you would like to edit by using either the Select All checkbox or individually selecting files. Next, click the "Edit Files" button above the file table and from the dropdown menu select if you would like to:

- Delete the selected files
- Edit the file metadata (file name, description) for the selected files
- Restrict the selected files
- Unrestrict the selected files (only if the selected files are restricted)
- Add tags to the selected files

You will not have to leave the dataset page to complete these action, except for editing file metadata, which will bring you to the Edit Files page. There you will have to click the “Save Changes” button to apply your edits and return to the dataset page.

If you restrict files, you will also prompted with a popup asking you to fill out the Terms of Access for the files. If Terms of Access already exist, you will be asked to confirm them. Note that some Dataverse installations do not allow for file restrictions.

### Edit File Variable Metadata

Variable Metadata can be edited directly through an API call (*API Guide: Editing Variable Level Metadata*) or by using the [Dataverse Data Curation Tool](#).

### File Path

The File Path metadata field is Dataverse’s way of representing a file’s location in a folder structure. When a user uploads a .zip file containing a folder structure, Dataverse automatically fills in the File Path information for each file contained in the .zip. If a user downloads the full dataset or a selection of files from it, they will receive a folder structure with each file positioned according to its File Path.











A file’s File Path can be manually added or edited on the Edit Files page. Changing a file’s File Path will change its location in the folder structure that is created when a user downloads the full dataset or a selection of files from it.

If there is more than one file in the dataset, and once at least one of them has a non-empty directory path, the Dataset Page will present an option for switching between the traditional table view, and the tree-like view of the files showing the folder structure, as in the example below:

## Change View

Table

Tree

- ▼  codebooks
  -  [codebook.pdf](#) (50.5 KB)
  -  [format.txt](#) (1.9 KB)
- ▼  data
  - ▼  aggregate
    -  [aggregates.tab](#) (40 B)
  - ▼  raw
    -  [output\\_data.tab](#) (126.7 KB)
    -  [quality\\_data.tab](#) (121.7 KB)
-  [README.txt](#) (109 B)

### File Tags

File tags are comprised of custom, category (i.e. Documentation, Data, Code) and tabular data tags (i.e. Event, Genomics, Geospatial, Network, Panel, Survey, Time Series). Use the dropdown select menus as well as the custom file tag input to apply these tags to the selected files. There is also a Delete Tags feature that, if checked, will allow you to delete unused file tags within that dataset.

### 1.4.6 Replace Files

In cases where you would like to revise an existing file rather than add a new one, you can do so using our Replace File feature. This will allow you to track the history of this file across versions of your dataset, both before and after replacing it. This could be useful for updating your data or fixing mistakes in your data. Because replacing a file creates an explicit link between the previous dataset version and the current version, the file replace feature is not available for unpublished dataset drafts. Also note that replacing a file will not automatically carry over that file's metadata, but once the file is replaced then its original metadata can still be found by referencing the previous version of the file under the "Versions" tab of the file page.

To replace a file, go to the file page for that file, click on the "Edit" button, and from the dropdown list select "Replace". This will bring you to the Replace File page, where you can see the metadata for the most recently published version

of the file and you can upload your replacement file. Once you have uploaded the replacement file, you can edit its name, description, and tags. When you're finished, click the "Save Changes" button.

After successfully replacing a file, a new dataset draft version will be created. A summary of your actions will be recorded in the "Versions" tab on both the dataset page and file page. The Versions tab allows you to access all previous versions of the file across all previous versions of your dataset, including the old version of the file before you replaced it.

### 1.4.7 Terms

Dataset terms can be viewed and edited from the Terms tab of the dataset page, or under the Edit dropdown button of a Dataset. There, you can set up how users can use your data once they have downloaded it (CC0 waiver or custom Terms of Use), how they can access your data if you have files that are restricted (terms of access), and enable a Guestbook for your dataset so that you can track who is using your data and for what purposes. These are explained in further detail below:

#### CC0 Public Domain Dedication

By default, all new datasets created through Dataverse's web UI are given a [Creative Commons CC0 Public Domain Dedication](#).

The [Creative Commons](#) organization defines a number of [licenses](#) that allow copyright holders to release their intellectual property more openly, with fewer legal restrictions than standard copyright enforces. Each Creative Commons license typically specifies simple terms for how the IP must be used, reused, shared, and attributed. In addition to these licenses, Creative Commons also provides public domain tools that make it easy to dedicate IP to the public domain.

In the context of Dataverse, their [CC0 Public Domain Dedication](#) allows you to unambiguously waive all copyright control over your data in all jurisdictions worldwide. Data released with CC0 can be freely copied, modified, and distributed (even for commercial purposes) without violating copyright. In most parts of the world, factual data is exempt from copyright anyway, but applying CC0 removes all ambiguity and makes the legal copyright status of the data as clear as possible. Dataverse applies CC0 to datasets by default because it facilitates reuse, extensibility, and long-term preservation of research data by assuring that the data can be safely handled by anyone without fear of potential copyright pitfalls.

Though CC0 waives a dataset owner's legal copyright controls over the data, it does not exempt Dataverse users from following ethical and professional norms in scholarly communications. The [Dataverse Community Norms](#) \* as well as scientific best practices assert that proper credit should be given via citation. Regardless of whether CC0 has been applied or not, Dataverse users are expected to cite the data they use, giving credit to the data's authors. This expectation applies to both the Dataverse Community and the entire wider scholarly community.

Additionally, users are still expected to respect access restrictions and other terms applied to CC0 files in Dataverse. Additional restrictions, conditions, and terms can still be compatible with CC0, as CC0 only operates in the realm of copyright, which is rather limited when it comes to data.

If a data owner feels that CC0 is not suitable for their data, they are able to enter custom Terms of Use, as detailed in the following section.

\* **Legal Disclaimer:** these [Community Norms](#) are not a substitute for the CC0 waiver or custom terms and licenses applicable to each dataset. The Community Norms are not a binding contractual agreement, and that downloading datasets from Dataverse does not create a legal obligation to follow these policies.

#### Custom Terms of Use for Datasets

If you are unable to use the CC0 Public Domain Dedication for your datasets, you may specify your own custom Terms of Use. To do so, select "No, do not apply CC0 - "Public Domain Dedication", and a Terms of Use text box

will show up allowing you to enter your own custom terms of use for your dataset. To add more information about the Terms of Use, we have provided fields like Special Permissions, Restrictions, Citation Requirements, etc.

Here is an [example of a Data Usage Agreement](#) for datasets that have de-identified human subject data.

### Restricted Files + Terms of Access

If you restrict any files in your dataset, you will be prompted by a pop-up to enter Terms of Access for the data. This can also be edited in the Terms tab or selecting Terms in the “Edit” dropdown button in the dataset. You may also allow users to request access for your restricted files by enabling “Request Access”. To add more information about the Terms of Access, we have provided fields like Data Access Place, Availability Status, Contact for Access, etc.

**Note:** Some Dataverse installations do not allow for file restriction.

### Guestbook

This is where you will enable a particular Guestbook for your dataset, which is setup at the Dataverse-level. For specific instructions please visit the [Dataset Guestbooks](#) section of the Dataverse Management page.

## 1.4.8 Roles & Permissions

Dataverse user accounts can be granted roles that define which actions they are allowed to take on specific dataverses, datasets, and/or files. Each role comes with a set of permissions, which define the specific actions that users may take.

Roles and permissions may also be granted to groups. Groups can be defined as a collection of Dataverse user accounts, a collection of IP addresses (e.g. all users of a library’s computers), or a collection of all users who log in using a particular institutional login (e.g. everyone who logs in with a particular university’s account credentials).

### Dataset-Level

Admins or curators of a dataset can assign roles and permissions to the users of that dataset. If you are an admin or curator of a dataset, then you can get to the dataset permissions page by clicking the “Edit” button, highlighting “Permissions” from the dropdown list, and clicking “Dataset”.

When you access a dataset’s permissions page, you will see two sections:

**Users/Groups:** Here you can assign roles to specific users or groups, determining which actions they are permitted to take on your dataset. You can also reference a list of all users who have roles assigned to them for your dataset and remove their roles if you please. Some of the users listed may have roles assigned at the dataverse level, in which case those roles can only be removed from the dataverse permissions page.

**Roles:** Here you can reference a full list of roles that can be assigned to users of your dataset. Each role lists the permissions that it offers.

### File-Level

If specific files in your dataset are restricted access, then you can grant specific users or groups access to those files while still keeping them restricted to the general public. If you are an admin or curator of a dataset, then you can get to the file-level permissions page by clicking the “Edit” button, highlighting “Permissions” from the dropdown list, and clicking “File”.

When you access a dataset’s file-level permissions page, you will see two sections:

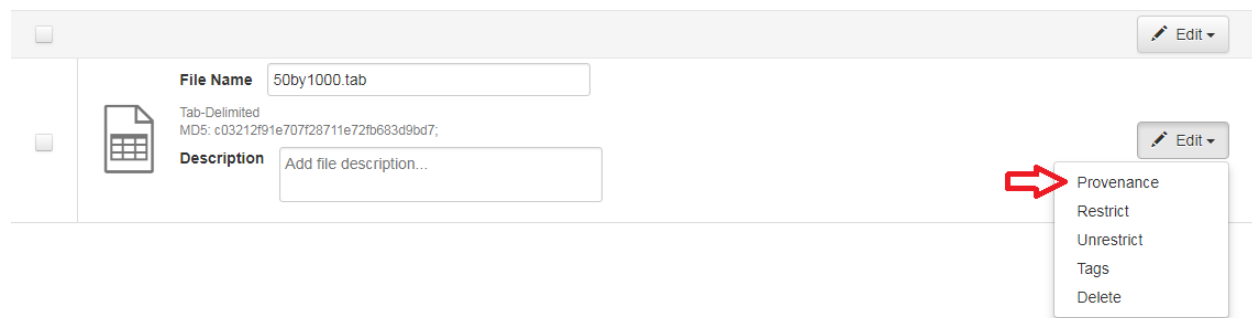
**Users/Groups:** Here you can see which users or groups have been granted access to which files. You can click the “Grant Access to Users/Groups” button to see a box where you can grant access to specific files within your dataset to specific users or groups. If any users have requested access to a file in your dataset, you can grant or reject their access request here.

**Restricted Files:** In this section, you can see the same information, but broken down by each individual file in your dataset. For each file, you can click the “Assign Access” button to see a box where you can grant access to that file to specific users or groups.

### 1.4.9 Data Provenance

Data Provenance is a record of where your data came from and how it reached its current form. It describes the origin of a data file, any transformations that have been made to that file, and any persons or organizations associated with that file. A data file’s provenance can aid in reproducibility and compliance with legal regulations. Dataverse can help you keep track of your data’s provenance. Currently, Dataverse only makes provenance information available to those who have edit permissions on your dataset, but in the near future we plan to expand this feature to make provenance information available to the public. You can track our progress in [this issue](#) on the Dataverse GitHub repository.

Dataverse accepts provenance information in two forms: a *Provenance File* or a free-text *Provenance Description*. You can attach this provenance information to your data files in Dataverse as part of the file upload process, by clicking Edit -> Provenance:



This will open a window where you can add your Provenance File and/or Provenance Description:



**Provenance** ✕

Provenance is a record of the origin of your data file and any transformations it has been through. Upload a JSON file from a provenance capture tool to generate a graph of your data's provenance. For more information, please refer to our [User Guide](#).

**Provenance File**      File must be JSON format and follow W3C standards.

+ Select File

You may also add information documenting the history of your data file, including how it was created, how it has changed, and who has worked with it.

**Provenance Description**      Add provenance description...

Save Changes      Cancel

A **Provenance File** is the preferred way of submitting provenance information to Dataverse because it provides a detailed and trustworthy record. Provenance files are typically generated during the process of data analysis, using provenance capture tools like provR, RDataTracker, NoWorkFlow, recordr, or CamFlow.

Once you upload a provenance file, Dataverse will need some additional information in order to accurately connect it to your data file. Once provenance file upload finishes, an input box labeled “Connect entity” will appear under the file. Provenance files contain a list of “entities”, which include your data file as well as any objects associated with it (e.g. a chart, a spellchecker, etc.). You will need to tell Dataverse which entity within the provenance file represents your data file. You may type the name of the entity into the box, or click the arrow next to the box and select the entity from a list of all entities in the provenance file.

For more information on entities and the contents of provenance files, see [the W3C PROV Model Primer](#).

Once you’ve uploaded your Provenance File and connected the proper entity, you can hit the Preview button to view the raw JSON of the Provenance File. This can help you confirm that you’ve uploaded the right file. Be sure to double-check it, because the Provenance File will be made *permanent* once it’s finalized. At that point you will not be able to *replace*, *remove*, or otherwise *edit* the Provenance File. This ensures that the Provenance File maintains a stable, immutable record of the data file’s history. This finalization of the Provenance File happens at different points depending on the status of your data file. If this is a brand new data file that has never been published before, then its associated Provenance File will be made permanent once you publish the dataset. If this data file *has* been published in a previous version of your dataset, then its associated Provenance File will be made permanent as soon as you upload the Provenance File and click “Save Changes” on the warning popup.

A **Provenance Description** allows you to add more provenance information in addition to or in place of a provenance file. This is a free-text field that allows you to enter any information you feel might be relevant to those interested

in learning about the provenance of your data. This might be a good place to describe provenance factors like what operating system you used when working with the data file, what functions or libraries you used, how data was merged into the file, what version of the file you used, etc. The Provenance Description is not as useful or trustworthy as a provenance file, but it can still provide value. Unlike the Provenance File, the Provenance Description is never made permanent: you can always edit, remove, or replace it at any time.

You can return to attach provenance to your data file later on by clicking the “Add + Edit Metadata” button on the file page, and then clicking the “Edit -> Provenance” button.

### 1.4.10 Thumbnails + Widgets

#### Thumbnails

Thumbnail images can be assigned to a dataset manually or automatically. The thumbnail for a dataset appears on the search result card for that dataset and on the dataset page itself. If a dataset contains one or more data files that Dataverse recognizes as an image, then one of those images is automatically selected as the dataset thumbnail.

If you would like to manually select your dataset’s thumbnail, you can do so by clicking the “Edit” button on your dataset, and selecting “Thumbnails + Widgets” from the dropdown menu.

On this page, under the Thumbnail tab you will see three possible actions.

**Select Available File:** Click the “Select Thumbnail” button to choose an image from your dataset to use as the dataset thumbnail.

**Upload New File:** Upload an image file from your computer to use as the dataset thumbnail. While by default your thumbnail image is drawn from a file in your dataset, this will allow you to upload a separate image file to use as your dataset thumbnail. This uploaded image file will only be used as the dataset thumbnail; it will not be stored as a data file in your dataset.

**Remove Thumbnail:** If you click the “Remove” button under the thumbnail image, you will remove the dataset’s current thumbnail. The Dataset will then revert to displaying a basic default icon as the dataset thumbnail.

When you’re finished on this page, be sure to click “Save Changes” to save what you’ve done.

Note: If you prefer, it is also possible to set an image file in your dataset as your thumbnail by selecting the file, going to Edit Files -> Metadata, and using the “Set Thumbnail” button.

#### Widgets

The Widgets feature provides you with code for your personal website so your dataset can be displayed. There are two types of Widgets for a dataset: the Dataset Widget and the Dataset Citation Widget. Widgets are found by going to your dataset page, clicking the “Edit” button (the one with the pencil icon) and selecting “Thumbnails + Widgets” from the dropdown menu.

In the Widgets tab, you can copy and paste the code snippets for the widget you would like to add to your website. If you need to adjust the height of the widget on your website, you may do so by editing the `heightPx=500` parameter in the code snippet.

#### Dataset Widget

The Dataset Widget allows the citation, metadata, files and terms of your dataset to be displayed on your website. When someone downloads a data file in the widget, it will download directly from the datasets on your website. If a file is restricted, they will be directed to your dataverse to log in, instead of logging in through the widget on your site.

To edit your dataset, you will need to return to the Dataverse repository where the dataset is stored. You can easily do this by clicking on the link that says “Data Stored in (Name) Dataverse” found in the bottom of the widget.

### **Dataset Citation Widget**

The Dataset Citation Widget will provide a citation for your dataset on your personal or project website. Users can download the citation in various formats by using the Cite Data button. The persistent URL in the citation will direct users to the dataset in your dataverse.

### **Adding Widgets to an OpenScholar Website**

1. Log in to your OpenScholar website
2. Either build a new page or navigate to the page you would like to use to show the Dataverse widgets.
3. Click on the Settings Cog and select Layout
4. At the top right, select Add New Widget and under Misc. you will see the Dataverse Dataset and the Dataverse Dataset Citation Widgets. Click on the widget you would like to add, fill out the form, and then drag it to where you would like it to display in the page.

#### **1.4.11 Publish Dataset**

When you publish a dataset (available to an Admin, Curator, or any custom role which has this level of permission assigned), you make it available to the public so that other users can browse or search for it. Once your dataset is ready to go public, go to your dataset page and click on the “Publish” button on the right hand side of the page. A pop-up will appear to confirm that you are ready to actually Publish since once a dataset is made public it can no longer be unpublished.

Whenever you edit your dataset, you are able to publish a new version of the dataset. The publish dataset button will reappear whenever you edit the metadata of the dataset or add a file.

Note: Prior to publishing your dataset the Data Citation will indicate that this is a draft but the “DRAFT VERSION” text will be removed as soon as you Publish.

#### **1.4.12 Submit for Review**

If you have a Contributor role (can edit metadata, upload files, and edit files, edit Terms, Guestbook, and Submit datasets for review) in a Dataverse you can submit your dataset for review when you have finished uploading your files and filling in all of the relevant metadata fields. To Submit for Review, go to your dataset and click on the “Submit for Review” button, which is located next to the “Edit” button on the upper-right. Once Submitted for Review: the Admin or Curator for this Dataverse will be notified to review this dataset before they decide to either “Publish” the dataset or “Return to Author”. If the dataset is published the contributor will be notified that it is now published. If the dataset is returned to the author, the contributor of this dataset will be notified that they need to make modifications before it can be submitted for review again.

#### **1.4.13 Private URL to Review Unpublished Dataset**

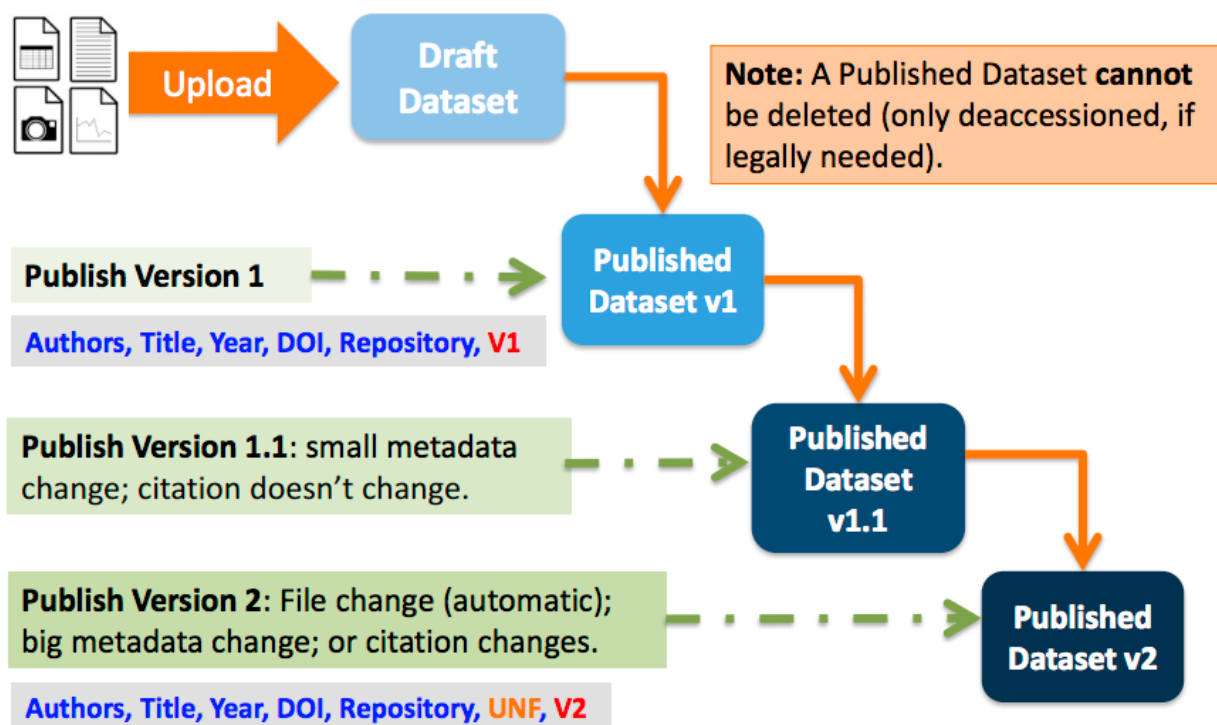
Creating a Private URL for your dataset allows you to share your dataset (for viewing and downloading of files) before it is published to a wide group of individuals who may not have a user account on Dataverse. Anyone you send the Private URL to will not have to log into Dataverse to view the dataset.

1. Go to your unpublished dataset
2. Select the “Edit” button
3. Select “Private URL” in the dropdown menu
4. In the pop-up select “Create Private URL”
5. Copy the Private URL which has been created for this dataset and it can now be shared with anyone you wish to have access to view or download files in your unpublished dataset.

To disable a Private URL and to revoke access, follow the same steps as above until step #3 when you return to the popup, click the “Disable Private URL” button.

### 1.4.14 Dataset Versions

Versioning is important for long-term research data management where metadata and/or files are updated over time. It is used to track any metadata or file changes (e.g., by uploading a new file, changing file metadata, adding or editing metadata) once you have published your dataset.



Once you edit your published dataset a new draft version of this dataset will be created. To publish this new version of your dataset, select the “Publish Dataset” button on the top right side of the page. If you were at version 1 of your dataset, depending on the types of changes you had made, you would be asked to publish your draft as either version 1.1 or version 2.0.

**Important Note:** If you add a file, your dataset will automatically be bumped up to a major version (e.g., if you were at 1.0 you will go to 2.0).

On the Versions tab of a dataset page, there is a versions table that displays the version history of the dataset. You can use the version number links in this table to navigate between the different versions of the dataset, including the unpublished draft version, if you have permission to access it.

There is also a Versions tab on the file page. The versions table for a file displays the same information as the dataset, but the summaries are filtered down to only show the actions related to that file. If a new dataset version were created without any changes to an individual file, that file's version summary for that dataset version would read "No changes associated with this version".

### Version Details

To view exactly what has changed, starting from the originally published version to any subsequent published versions: click the Versions tab on the dataset page to see all versions and changes made for that particular dataset.

Once you have more than one version (this can simply be version 1 and a draft), you can click the "View Details" link next to each summary to learn more about the metadata fields and files that were either added or edited. You can also click the checkboxes to select any two dataset versions, then click the "View Differences" button to open the Version Differences Details popup and compare the differences between them.

### 1.4.15 Dataset Metrics and Make Data Count

All installations of Dataverse count file downloads. These file download counts are aggregated and reported at the Dataset level as well as at the file level.

Some installations of Dataverse also have support for expanded metrics at the dataset level for views, file downloads, and citations using Make Data Count standards. [Make Data Count](#) is a project to collect and standardize metrics on data use, especially views, downloads, and citations. Citations for datasets are retrieved from [Crossref](#) via DataCite using Make Data Count standards.

For the specific API calls for Make Data Count, see [Dataset Metrics](#) in the [Native API](#) section of the API Guide.

### 1.4.16 Cloud Storage + Computing

Dataverse installations can be configured to facilitate cloud-based storage and/or computing (this feature is considered experimental at this time, and some of the kinks are still being worked out). While the default configuration for Dataverse uses a local file system for storing data, a cloud-enabled Dataverse installation can use a Swift object storage database for its data. This allows users to perform computations on data using an integrated cloud computing environment.

#### Cloud Computing

The "Compute" button on dataset and file pages will allow you to compute on a single dataset, multiple datasets, or a single file. You can use it to build a compute batch and go directly to the cloud computing environment that is integrated with Dataverse.

#### Cloud Storage Access

If you need to access a dataset in a more flexible way than the Compute button provides, then you can use the Cloud Storage Access box on the dataset page to copy the dataset's container name. This unique identifier can then be used to allow direct access to the dataset.

## 1.4.17 Dataset Deaccession

**Warning:** It is not recommended that you deaccession a dataset or a version of a dataset. This is a very serious action that should only occur if there is a legal or valid reason for the dataset to no longer be accessible to the public. If you absolutely must deaccession, you can deaccession a version of a dataset or an entire dataset.

To deaccession, go to your published dataset (or add a new one and publish it), click the “Edit” button, and from the dropdown menu select “Deaccession Dataset”. If you have multiple versions of a dataset, you can select here which versions you want to deaccession or choose to deaccession the entire dataset.

You must also include a reason as to why this dataset was deaccessioned. Select the most appropriate reason from the dropdown list of options. If you select “Other”, you must also provide additional information.

Add more information as to why this was deaccessioned in the free-text box. If the dataset has moved to a different repository or site you are encouraged to include a URL (preferably persistent) for users to continue to be able to access this dataset in the future.

If you deaccession the most recently published version of the dataset but not all versions of the dataset, you may then revisit an earlier version and create a new non-deaccessioned draft for the dataset. For example, imagine you have a version 1 and version 2 of a dataset, both published, and you deaccession version 2. You may then edit version 1 of the dataset and a new draft version will be created.

**Important Note:** A tombstone landing page with the basic citation metadata will always be accessible to the public if they use the persistent URL (Handle or DOI) provided in the citation for that dataset. Users will not be able to see any of the files or additional metadata that were previously available prior to deaccession.

## 1.5 Tabular Data File Ingest

Contents:

### 1.5.1 Supported File Formats

Tabular Data ingest supports the following file formats:

File format	Versions supported
SPSS (POR and SAV formats)	7 to 22
STATA	4 to 13
R	up to 3
Excel	XLSX only (XLS is NOT supported)
CSV (comma-separated values)	(limited support)

See the subsections in the left sidebar for more information on each of these supported formats.

### 1.5.2 Tabular Data, Representation, Storage and Ingest

This section explains the basics of how tabular data is handled in the application and what happens during the ingest process, as the files uploaded by the user are processed and converted into the archival format in the Dataverse application.

**Contents:**

- *What Happens During this “Ingest”?*
- *Tabular Data and Metadata*
  - *Data vs. Metadata*
  - *Tabular Metadata in Dataverse*

**What Happens During this “Ingest”?**

The goal of our ingest process is to extract the data content from the user’s files and archive it in an application-neutral, easily-readable format. What does this mean? - Commercial applications such as SPSS and Stata use their own, proprietary formats to encode their files. Some companies publish the specifications of their formats (Thank you Stata - much appreciated!), some don’t (SPSS - yes, we are still frowning at you here at the Dataverse Project). Either way, reading these specially-formatted files requires some extra knowledge or special software. For these reasons they are not considered ideal for the purposes of archival preservation. Dataverse stores the raw data content extracted from such files in plain text, TAB-delimited files. The metadata information that describes this content is stored separately, in a relational database, so that it can be accessed efficiently by the application. For the purposes of archival preservation it can be exported, in plain text XML files, using a standardized, open [DDI Codebook](#) format. (more info below)

**Tabular Data and Metadata****Data vs. Metadata**

A simple example is a numeric data column in a user’s Stata file that contains 0s and 1s. These numeric values will be extracted and stored in a TAB-delimited file. By themselves, if you don’t know what these values represent, these 1s and 0s are not meaningful data. So the Stata file has some additional information that describes this data vector: it represents the observation values of a *variable* with the *name* “party”; with a descriptive *label* “Party Affiliation”; and the 2 numeric values have *categorical labels* of “Democrat” for 0 and “Republican” for 1. This extra information that adds value to the data is *metadata*.

**Tabular Metadata in Dataverse**

The structure of the metadata defining tabular data variables used in Dataverse was originally based on the [DDI Codebook](#) format.

You can see an example of DDI output under the “Data Variable Metadata Access” section of the [Data Access API](#) section of the API Guide.

**1.5.3 SPSS**

SPSS data files (POR and SAV formats).

**Contents:**

- *Supported Versions*
- *Limitations*

- *SPSS Control Cards - not supported*
- *Support for Language Encodings in SPSS*

### Supported Versions

Dataverse supports reading of all SPSS versions 7 to 22. But please see the “Limitations” section.

### Limitations

SPSS does not openly publish the specifications of their proprietary file formats. Our ability to read and parse their files is based on some documentation online from unofficial sources, and some reverse engineering. Because of that we cannot, unfortunately, guarantee to be able to process *any* SPSS file uploaded.

However, we’ve been improving this process for a few years by now, and it should be quite robust in the current version of Dataverse. Thus your chances of success - uploading an SPSS files and having it turned into a fully functional tabular data table in the Dataverse - should be reasonably good.

If you are having a problem with a particular SPSS file, please contact our support and we’ll see if it’s something we could further improve in our SPSS ingest driver.

### SPSS Control Cards - not supported

In the past, there have been attempts to support SPSS “control cards”, in addition to the .SAV and .POR files; both in the early VDC software, and in some versions of DVN. A “control card” is a plain text file with a set of SPSS commands that describe the raw data, provided in a separate, fixed-width-columns or comma-separated-values file. For various reasons, it has never been very successful. We weren’t seeing too much demand for this ingest feature, so it was dropped from Dataverse 4.0.

Please contact us if you have any questions and/or strong feelings on this issue, or if you have serious amounts of data exclusively available in this format that you would like to ingest under Dataverse.

### Support for Language Encodings in SPSS

Historically, there was no support for specifying a particular language/code page encoding for the data stored in an SPSS file. Meaning, text values in none-ASCII encodings, or non-Latin characters could be entered and stored, but there was no setting to unambiguously specify what language, or what character set it was. By default, Dataverse will try to interpret binary characters as UTF8. If that’s not working - for example, if the descriptive labels and/or categorical values ingest as garbage - and if you happen to know what encoding was used in the original file, you can now specify it in the Ingest Options.

For example, if you know that the text in your SAV file is in Mandarin, and is encoded using the GB2312, specify it as follows:

Upload your file, in the “Edit Files” tab of the Dataset page. Once the file is recognized as SPSS/save, and *before* you click Save, go into the “Advanced Ingest Options”, and select “Simplified Chinese, GB2312” in the nested menu under “Language Encoding” -> “East Asian”.



## 1.5.4 Stata

Of all the third party statistical software providers, Stata does the best job at documenting the internal format of their files, by far. And at making that documentation freely and easily available to developers (yes, we are looking at you, SPSS). Because of that, Stata is the best supported format for tabular data ingest.

**New in Dataverse 4.0:** support for Stata v.13 has been added.

## 1.5.5 R Data Format

Support for R (.RData) files has been introduced in DVN 3.5.

### Contents:

- *Overview.*
- *Data Formatting Requirements.*
- *Data Types, compared to other supported formats (Stat, SPSS)*
  - *Integers, Doubles, Character strings*
  - *R Factors*
  - *Boolean values*
  - *Limitations of R, as compared to SPSS and STATA.*
- *Time values in R*

### Overview.

R has been increasingly popular in the research/academic community, owing to the fact that it is free and open-source (unlike SPSS and STATA). Consequently, there is an increasing amount of data available exclusively in R format.

### Data Formatting Requirements.

The data must be formatted as an R dataframe (`data.frame()`). If an .RData file contains multiple dataframes, only the 1st one will be ingested (this may change in the future).

### Data Types, compared to other supported formats (Stat, SPSS)

#### Integers, Doubles, Character strings

The handling of these types is intuitive and straightforward. The resulting tab file columns, summary statistics and UNF signatures should be identical to those produced by ingesting the same vectors from SPSS and Stata.

#### Things that are unique to R:

R explicitly supports Missing Values for all of the types above; Missing Values encoded in R vectors will be recognized and preserved in TAB files, counted in the generated summary statistics and data analysis. Please note however that the Dataverse notation for a missing value, as stored in a TAB file, is an empty string, an not “NA” as in R.

In addition to Missing Values, R recognizes “Not a Value” (NaN) and positive and negative infinity for floating point variables. These are now properly supported by the Dataverse.

Also note, that unlike Stata, that does recognize “float” and “double” as distinct data types, all floating point values in R are in fact doubles.

### R Factors

These are ingested as “Categorical Values” in the Dataverse.

One thing to keep in mind: in both Stata and SPSS, the actual value of a categorical variable can be both character and numeric. In R, all factor values are strings, even if they are string representations of numbers. So the values of the resulting categoricals in the Dataverse will always be of string type too.

Another thing to note is that R factors have no builtin support for SPSS or STATA-like descriptive labels. This is in fact potentially confusing, as they also use the word “label”, in R parlance. However, in the context of a factor in R, it still refers to the “payload”, or the data content of its value. For example, if you create a factor with the “labels” of *democrat*, *republican* and *undecided*, these strings become the actual values of the resulting vector. Once ingested in the Dataverse, these values will be stored in the tab-delimited file. The Dataverse `DataVariable` object representing the vector will be of type “Character” and have 3 `VariableCategory` objects with the *democrat*, etc. for **both** the `CategoryValue` and `CategoryLabel`. (In one of the future releases, we are planning to make it possible for the user to edit the `CategoryLabel`, using it for its intended purpose - as a descriptive, human-readable text text note).

To properly handle R vectors that are *ordered factors* Dataverse (starting with DVN 3.6) supports the concept of an “Ordered Categorical” - a categorical value where an explicit order is assigned to the list of value labels.

### Boolean values

R Boolean (logical) values are supported.

### Limitations of R, as compared to SPSS and STATA.

Most noticeably, R lacks a standard mechanism for defining descriptive labels for the data frame variables. In the Dataverse, similarly to both Stata and SPSS, variables have distinct names and labels; with the latter reserved for longer, descriptive text. With variables ingested from R data frames the variable name will be used for both the “name” and the “label”.

*Optional R packages exist for providing descriptive variable labels; in one of the future versions support may be added for such a mechanism. It would of course work only for R files that were created with such optional packages.*

Similarly, R categorical values (factors) lack descriptive labels too. **Note:** This is potentially confusing, since R factors do actually have “labels”. This is a matter of terminology - an R factor’s label is in fact the same thing as the “value” of a categorical variable in SPSS or Stata and Dataverse; it contains the actual meaningful data for the given observation. It is NOT a field reserved for explanatory, human-readable text, such as the case with the SPSS/Stata “label”.

Ingesting an R factor with the level labels “MALE” and “FEMALE” will produce a categorical variable with “MALE” and “FEMALE” in the values and labels both.

### Time values in R

This warrants a dedicated section of its own, because of some unique ways in which time values are handled in R.

R makes an effort to treat a time value as a real time instance. This is in contrast with either SPSS or Stata, where time value representations such as “Sep-23-2013 14:57:21” are allowed; note that in the absence of an explicitly defined time zone, this value cannot be mapped to an exact point in real time. R handles times in the “Unix-style” way: the

value is converted to the “seconds-since-the-Epoch” Greenwich time (GMT or UTC) and the resulting numeric value is stored in the data file; time zone adjustments are made in real time as needed.

Things still get ambiguous and confusing when R **displays** this time value: unless the time zone was explicitly defined, R will adjust the value to the current time zone. The resulting behavior is often counter-intuitive: if you create a time value, for example:

```
timevalue<-as.POSIXct("03/19/2013 12:57:00", format = "%m/%d/%Y %H:%M:%OS");
```

on a computer configured for the San Francisco time zone, the value will be differently displayed on computers in different time zones; for example, as “12:57 PST” while still on the West Coast, but as “15:57 EST” in Boston.

If it is important that the values are always displayed the same way, regardless of the current time zones, it is recommended that the time zone is explicitly defined. For example:

```
attr(timevalue, "tzzone")<-"PST"
```

or

```
timevalue<-as.POSIXct("03/19/2013 12:57:00", format = "%m/%d/%Y %H:%M:%OS",
tz="PST");
```

Now the value will always be displayed as “15:57 PST”, regardless of the time zone that is current for the OS ... **BUT ONLY** if the OS where R is installed actually understands the time zone “PST”, which is not by any means guaranteed! Otherwise, it will **quietly adjust** the stored GMT value to **the current time zone**, yet it will still display it with the “PST” tag attached!\*\* One way to rephrase this is that R does a fairly decent job **storing** time values in a non-ambiguous, platform-independent manner - but gives you no guarantee that the values will be displayed in any way that is predictable or intuitive.

In practical terms, it is recommended to use the long/descriptive forms of time zones, as they are more likely to be properly recognized on most computers. For example, “Japan” instead of “JST”. Another possible solution is to explicitly use GMT or UTC (since it is very likely to be properly recognized on any system), or the “UTC+<OFFSET>” notation. Still, none of the above **guarantees** proper, non-ambiguous handling of time values in R data sets. The fact that R **quietly** modifies time values when it doesn’t recognize the supplied timezone attribute, yet still appends it to the **changed** time value does make it quite difficult. (These issues are discussed in depth on R-related forums, and no attempt is made to summarize it all in any depth here; this is just to made you aware of this being a potentially complex issue!)

An important thing to keep in mind, in connection with the Dataverse ingest of R files, is that it will **reject** an R data file with any time values that have time zones that we can’t recognize. This is done in order to avoid (some) of the potential issues outlined above.

It is also recommended that any vectors containing time values ingested into the Dataverse are reviewed, and the resulting entries in the TAB files are compared against the original values in the R data frame, to make sure they have been ingested as expected.

Another **potential issue** here is the **UNF**. The way the UNF algorithm works, the same date/time values with and without the timezone (e.g. “12:45” vs. “12:45 EST”) **produce different UNFs**. Considering that time values in Stata/SPSS do not have time zones, but ALL time values in R do (yes, they all do - if the timezone wasn’t defined explicitly, it implicitly becomes a time value in the “UTC” zone!), this means that it is **impossible** to have 2 time value vectors, in Stata/SPSS and R, that produce the same UNF.

**A pro tip:** if it is important to produce SPSS/Stata and R versions of the same data set that result in the same UNF when ingested, you may define the time variables as **strings** in the R data frame, and use the “YYYY-MM-DD HH:mm:ss” formatting notation. This is the formatting used by the UNF algorithm to normalize time values, so doing the above will result in the same UNF as the vector of the same time values in Stata.

Note: date values (dates only, without time) should be handled the exact same way as those in SPSS and Stata, and should produce the same UNFs.

## 1.5.6 Excel

Excel files (**New** in Dataverse 4.0!)

Note: only the “new”, XLSX Excel files are supported. We are not planning to add support for the old-style, binary XLS files.

## 1.5.7 CSV/TSV

### Contents:

- *Ingest of Comma-Separated Values and Tab-Separated Values files as tabular data.*
- *Main formatting requirements*
- *Limitations*
- *Recognized data types and formatting*

### Ingest of Comma-Separated Values and Tab-Separated Values files as tabular data.

Dataverse will make an attempt to turn CSV and TSV files uploaded by the user into tabular data, using the [Apache CSV parser](#).

#### Main formatting requirements

The first row in the document will be treated as the CSV’s header, containing variable names for each column.

Each following row must contain the same number of comma-separated values (“cells”) as that header.

As of the Dataverse 4.8 release, we allow ingest of CSV files with commas and line breaks within cells. A string with any number of commas and line breaks enclosed within double quotes is recognized as a single cell. Double quotes can be encoded as two double quotes in a row (“”).

For example, the following lines:

```
a,b,"c,d  
efgh"ijk"l",m,n
```

are recognized as a **single** row with **5** comma-separated values (cells):

```
a  
b  
c,d\nefgh"ijk"l  
m  
n
```

(where \n is a new line character)

#### Limitations

Compared to other formats, relatively little information about the data (“variable-level metadata”) can be extracted from a CSV file. Aside from the variable names supplied in the top line, the ingest will make an educated guess about

the data type of each comma-separated column. One of the supported rich file formats (Stata, SPSS and R) should be used if you need to provide more descriptive variable-level metadata (variable labels, categorical values and labels, explicitly defined data types, etc.).

## Recognized data types and formatting

The application will attempt to recognize numeric, string, and date/time values in the individual columns.

For dates, the `yyyy-MM-dd` format is recognized.

For date-time values, the following 2 formats are recognized:

```
yyyy-MM-dd HH:mm:ss
```

```
yyyy-MM-dd HH:mm:ss z (same format as the above, with the time zone specified)
```

For numeric variables, the following special values are recognized:

`inf`, `+inf` - as a special IEEE 754 “positive infinity” value;

`NaN` - as a special IEEE 754 “not a number” value;

An empty value (i.e., a comma followed immediately by another comma, or the line end), or `NA` - as a *missing value*.

`null` - as a numeric *zero*.

(any combinations of lower and upper cases are allowed in the notations above).

In character strings, an empty value (a comma followed by another comma, or the line end) is treated as an empty string (NOT as a *missing value*).

Any non-Latin characters are allowed in character string values, **as long as the encoding is UTF8**.

**Note:** When the ingest recognizes a CSV or TSV column as a numeric vector, or as a date/time value, this information is reflected and saved in the database as the *data variable metadata*. To inspect that metadata, click on the *Download* button next to a tabular data file, and select *Variable Metadata*. This will export the variable records in the DDI XML format. (Alternatively, this metadata fragment can be downloaded via the Data Access API; for example: `http://localhost:8080/api/access/datafile/<FILEID>/metadata/ddi`).

The most immediate implication is in the calculation of the UNF signatures for the data vectors, as different normalization rules are applied to numeric, character, and date/time values. (see the *Universal Numerical Fingerprint (UNF)* section for more information). If it is important to you that the UNF checksums of your data are accurately calculated, check that the numeric and date/time columns in your file were recognized as such (as `type=numeric` and `type=character`, `category=date(time)`, respectively). If, for example, a column that was supposed to be numeric is recognized as a vector of character values (strings), double-check that the formatting of the values is consistent. Remember, a single improperly-formatted value in the column will turn it into a vector of character strings, and result in a different UNF. Fix any formatting errors you find, delete the file from the dataset, and try to ingest it again.

## 1.6 Data Exploration Guide

The installation of Dataverse you are using may have additional or different tools configured than what appears below. For the complete list of available tools, see *Inventory of External Tools*.

Contents:

## 1.6.1 TwoRavens: Tabular Data Exploration

**Please note:** This document is a bit old and may need an update. The TwoRavens project has a more recently published user guide on their site: <http://2ra.vn/papers/tworavens-guide.pdf>.

### Contents:

- *Exploring and Analyzing Tabular files in Dataverse*
- *Selection/Left Panel*
  - *Original Data and Subset Data*
  - *Variables, Models, and Summary*
- *Modeling/Center Panel*
- *Results/Right Panel*
  - *Subset*
  - *Set Covariates*
- *Additional Buttons*
  - *Estimate*
  - *Force*
  - *Reset*

### Exploring and Analyzing Tabular files in Dataverse

On the files tab, click on the “Explore” button to initiate TwoRavens Data Exploration and Analysis Tool.

#### Selection/Left Panel

The left panel contains two sets of buttons: (1) Original Data and Subset Data; and (2) Variables, Models, and Summary.

#### Original Data and Subset Data

When TwoRavens is initiated, you begin with the original dataset, and thus the Original Data button is selected. You will not be able to select the Subset Data until you have subsetted the data using the subset and select features in the right panel. After you have selected a subset of your data, you may toggle between that subset and the original data. If you wish to select a different subset, you may do so, but note that only one subset is supported at a time.

#### Variables, Models, and Summary

Each of these tabs displays something different in the left panel when selected. The Variables tab shows a list of all the variables. When a variable name is hovered over, you can see that variable’s summary statistics. The first three variables are selected by default and displayed in the center panel, but you may add or remove variables by clicking on their name in the Variables tab.

The Models tab displays a list of Zelig models that are supported by TwoRavens. A brief model description is visible when hovering on the model name. Depending on the level of measurement of the dependent variable (continuous, ordinal, dichotomous, etc.), particular models may or may not be appropriate.

Note that to estimate a model, you must select one from the list. Currently, please use only Ordinary Least Squares (ls) as we are working on making other models available. (Suggestion: maybe we need to gray out the ones the other ones for the time being)

The Summary tab shows summary statistics when a pebble is hovered over. If one removes the pointer from hovering over a pebble, the previous tab will be displayed. So, if you wish to have the summary tab displayed regardless of where the pointer is, click on Summary before hovering over a pebble. Otherwise, if Variables or Models has been the last tab selected, when the pointer is no longer hovering over a pebble, the table will return to Variables or Models.

## Modeling/Center Panel

The center panel displays a graphical representation of variable relations and denotes variables that have been tagged with certain properties. Variables may be tagged as either a dependent variable, a time series variable, or a cross sectional variable. Each of these are accomplished by clicking on the appropriate button at the top of the screen, and then clicking on a pebble in the center panel. You'll notice that when a variable is tagged with a property, the fill color becomes white, and the outline (or stroke) of the pebble turns the color of the property's button. Note that to estimate a model, the dependent variable must be selected.

Variable relations are specified by point-click-drag from one pebble to the other. When a path between pebbles has been specified, it is visually presented with a dotted arrow line and may be removed by pushing the delete key on your keyboard.

## Results/Right Panel

This section allows you to specify a subset of the data that you wish to estimate the model on, or that you wish to select and see updated summary statistics and distributions, and to set covariate values for the Zelig simulations (this is Zelig's setx function).

### Subset

To subset the data, click on the subset button and highlight (or brush) the portion of the distribution that you wish to use. You may remove the selection by clicking anywhere on the plot that is outside of the selected area. Or, if you wish to move the selected region, click inside the selection and move it to the left or right. If no region is selected, then by default the full range of values are used. If more than one variable is selected for subsetting, only the overlapping region is used in the model. If there are no overlapping regions, (i.e., if subsetted there would be no data), then only the first variable is used. Notice that range (or extent) of the selection for each variable is displayed below.

With a region selected, you have two options. First, you may click the Estimate button to estimate a model on using only the specified subset. Second, you may click the Select button. This will not estimate a model, but it will subset the data and return new summary statistics and plots for the subset. You may wish to use this feature to see how a subset will change the Set Covariate (Zelig's setx) defaults, for example. After selecting a subset, you may toggle back and forth between the subsetted data and the original data by activating the appropriate button in the left panel.

### Set Covariates

The Set Covariates button plots the distributions of each of the pebbles with an additional axis that contains two sliders, each of which default to the variable's mean. This is TwoRavens' equivalent of Zelig's setx function. Move these sliders to the left or right to set your covariates at the desired value prior to estimation. Notice that the selected values appear below each plot.

After clicking the Estimate button, the model will be estimated and, upon completion, results appear in the Results tab. The results include figures produced by Zelig (and eventually the equation that has been estimated, the R code used to estimate the model, and a results table).

### Additional Buttons

#### Estimate

This executes the specified statistical model. Notice the presence of blue highlight on the “Estimate” button while process is running, turning into green upon completion. Note: you cannot use estimate without selecting a dependent variable and a model.

#### Force

The Force button allows you to control the way layout of the pebbles. To use this feature, first make sure none of the pebbles are highlighted. If one is, simply click on it to remove the highlighting. Second, press and hold the control key. Third, while holding down the control key, click the Force button. Fourth, continue to hold the control key and click on a pebble. You may now release the control key. Click on a pebble and drag it around on your screen.

#### Reset

This is your start over button. Clicking this is equivalent to reloading the Web page or re-initiating TwoRavens.

## 1.6.2 WorldMap: Geospatial Data Exploration

### Contents:

- *Dataverse and WorldMap*
- *What is Geoconnect?*
- *Mapping shapefiles with Geoconnect*
- *Mapping tabular files with Geoconnect*
  - *Preparing a tabular file to be mapped*
  - *Creating the map*
- *Finalizing your map*
- *Removing your map*

### Dataverse and WorldMap

WorldMap is developed by the Center for Geographic Analysis (CGA) at Harvard and is open source software that helps researchers visualize and explore their data in maps. The WorldMap and Dataverse collaboration allows researchers to upload shapefiles or tabular files to Dataverse for long term storage and receive a persistent identifier (through DOI), then easily navigate into WorldMap to interact with the data.

Note: WorldMap hosts their own [user guide](#) that covers some of the same material as this page.



## What is Geoconnect?

Geoconnect is a platform that integrates Dataverse and WorldMap, allowing researchers to visualize their geospatial data. Geoconnect can be used to create maps of shapefiles or of tabular files containing geospatial information. Geoconnect is an optional component of Dataverse, so if you are interested in this feature but don't see it in the installation of Dataverse you are using, you should contact the support team for that installation and ask them to enable the Geoconnect feature.

If a data file's owner has created a map of that data using Geoconnect, you can view the map by clicking the "Explore" button. If the data is in the form of a shapefile, the button takes you right to the map. If it's a tabular file, the Explore button will be a dropdown, and you'll need to select "Worldmap".

## Mapping shapefiles with Geoconnect

Geoconnect is capable of mapping shapefiles which are uploaded to Dataverse in .zip format. Specifically, Dataverse recognizes a zipped shapefile by:

1. Examining the contents of the .zip file
2. Checking for the existence of four similarly named files with the following extensions: .dbf, .prj, .shp, .shx

Once you have uploaded your .zip shapefile, a Map Data button will appear next to the file in the dataset. In order to use this button, you'll need to publish your dataset. Once your dataset has been published, you can click on the Map Data button to be brought to Geoconnect, the portal between Dataverse and WorldMap that will allow you to create your map.

To get started with visualizing your shapefile, click on the blue "Visualize on WorldMap" button in Geoconnect. It may take up to 45 seconds for the data to be sent to WorldMap and then back to Geoconnect.

Once this process has finished, you will be taken to a new page where you can style your map through Attribute, Classification Method, Number of Intervals, and Colors. Clicking "Apply Changes" will send your map to both Dataverse and WorldMap, creating a preview of your map that will be visible on your file page and your dataset page.

Clicking "View on WorldMap" will open WorldMap in a new tab, allowing you to see how your map will be displayed there.

You can delete your map with the "Delete" button. If you decide to delete the map, it will no longer appear on WorldMap, and your dataset in Dataverse will no longer display the map preview.

When you're satisfied with your map, you may click "Return to the Dataverse" to go back to Dataverse.

In the future, to replace your shapefile's map with a new one, simply click the Map Data button on the dataset or file page to return to the Geoconnect edit map page.

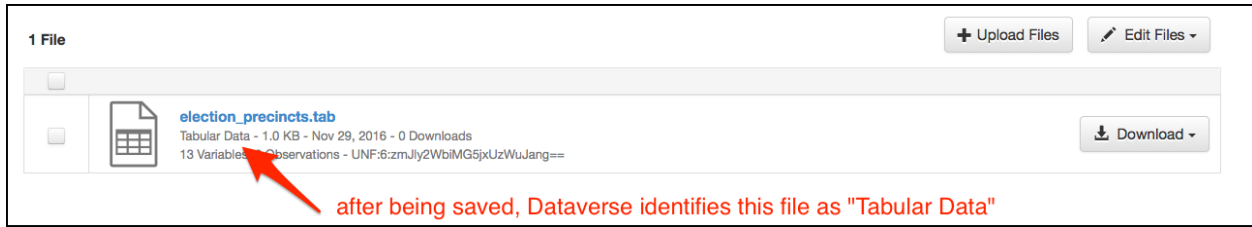
## Mapping tabular files with Geoconnect

Geoconnect can map tabular files that contain geospatial information such as latitude/longitude coordinates, census tracts, zip codes, Boston election wards, etc.

## Preparing a tabular file to be mapped

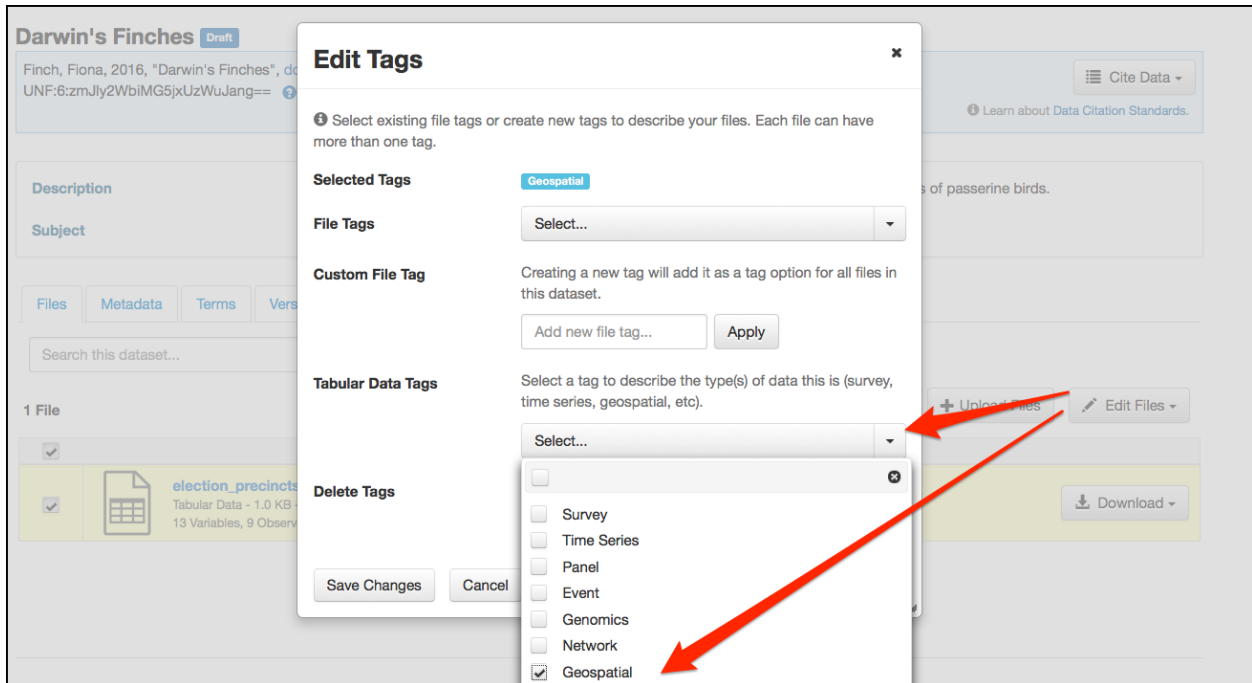
### 1. Ingest

Geospatial tabular files need a bit of preparation in Dataverse before they can be mapped in Geoconnect. When you upload your file, Dataverse will take about ten seconds to ingest it. During the ingest process it will identify the file as tabular data.



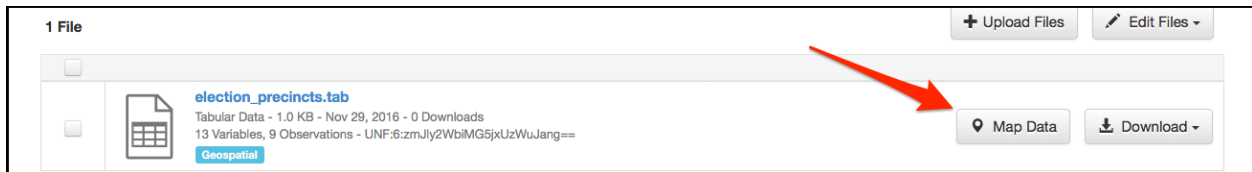
## 2. Tag as Geospatial

Next, you'll need to let Dataverse know that your tabular file contains geospatial data. Select your file, click the "Edit Files" button, and select "Tags" from the dropdown menu. This will take you to the Edit Tags menu (pictured below). Under the "Tabular Data Tags" dropdown, select "Geospatial". Then click "Save Changes".



## 3. Publish & Map Data

At this point, a "Map data" button will appear next to your file. Publish this new version of your dataset to activate this button.



## Creating the map

If your tabular file contains **latitude and longitude** columns, then the process is simple: those columns may be directly mapped. Otherwise, you will need to use a *spatial join*. Spatial joins tell WorldMap how to read your tabular data file in order to create a map that accurately represents it.

To carry out a spatial join, you'll manually connect

- **Geospatial column(s) from your Dataverse tabular file**

- e.g., a census tract column from your table

with

- A WorldMap “target layer” that contains the same geospatial information
  - e.g., WorldMap’s “target layer” containing census tract parameters

The following screenshots illustrate the mapping process:

**1. Once you’ve pressed the “Map Data” button, you’re brought to this page:**

Geoconnect: Map Data Set Up beta

**election\_precincts.tab**  
 Doe, John, 2017, "2010 Census Tract - MA", doi:10.5072/FK2/VZ1ZX9, DVN-Build 2 Dataverse, V2, UNF:6:z7latw5T7nTb4dBOAAvmAw==

Preview of your Dataverse file

Preview Data File

Here is a preview of the first few rows of your data file for reference when selecting the columns which contain geospatial data.

Displaying 5 of 13 rows

Row # ▲	animal ↕	election_precinct_int ↕	election_precinct_double ↕	neighborhood ↕	police_district_id ↕	public_works_id ↕	tiger_line_id ↕	road_left_size ↕
1	Dog	1	1.0	East Boston	1	2-06	85729227	205871733
2	Squirrel	3	3.0	Charlestown	2	1-09	85699791	205871735
3	Antelope	4	4.0	South Boston	3	1-1A	640323976	205871283
4	Zebra	6	6.0	Bronx	4	1-1B	85695847	258627915
5	Lion	7	7.0	Roslindale	5	2-04	637089796	257444575

Map Data File

Select the type of geospatial data contained in your data file, and the column(s) in which it can be found.

Geospatial Data      Type

Select...

Pick a Geospatial Data type

Cancel

**2. Choose a Geospatial Data Type**

Map Data File

Select the type of geospatial data contained in your data file, and the column(s) in which it can be found.

Geospatial Data      Type

Select...

- Select...
- Latitude/Longitude
- Boston, Administrative Geography**
- US Census Block
- US Census Block Group
- US Census Tract
- US Zip Code

List of Geospatial Data types

Cancel

**3. Choose a column from your file to match the WorldMap Layer you selected**

Map Data File

Select the type of geospatial data contained in your data file, and the column(s) in which it can be found.

**Geospatial Data**      **Type**      **Column Name**

Boston, Administrative Geography      Select...

Cancel      **Choose a column from your tabular data file** →

Select...  
 animal  
 election\_precinct\_int  
 election\_precinct\_double  
 neighborhood  
 police\_district\_id  
**public\_works\_id**  
 tiger\_line\_id  
 road\_left\_size  
 road\_right\_size  
 geoid10\_census\_tract  
 bg\_id\_10\_census\_block\_group  
 census\_block\_fips15  
 zip5

Developed at the Institute for Quantitative Social Science | Dataverse Project on [Twitter](#) | Code available at [GitHub](#)      2017 14:44:53  
 Copyright © 2017, The President & Fellows of Harvard College

**4. Choose from the list of WorldMap Layers available for the Geospatial Data Type you selected**

Map Data File

Select the type of geospatial data contained in your data file, and the column(s) in which it can be found.

**Geospatial Data**      **Type**      **Column Name**

Boston, Administrative Geography      public\_works\_id

Select the WorldMap layer to map your geospatial data onto.

**WorldMap Layer**      **Name**      **Description**

Select...      This is a brief description about the layer.

Submit Data to WorldMap

Select...  
 Select.....  
 2015 - Addresses, Boston  
 2015 - BRA Neighborhood Statistical Areas, Boston  
 2015 - BRA Planning Districts, Boston  
 2015 - Election Precincts, Boston  
 2015 - Election Wards, Boston  
 2015 - Fire Districts, Boston  
 2015 - Intersections (Census), Boston  
 2015 - Intersections (City), Boston  
 2015 - ISD Neighborhoods (Permits), Boston  
 2015 - Police Districts, Boston  
**2015 - Public Works Districts, Boston**  
 2015 - Roads, Boston  
 2015 - Roads (left side), Boston  
 2015 - Roads (right side), Boston

Developed at the Institute for Quantitative Social Science | Dataverse Project on [Twitter](#) | Code available at [GitHub](#)      April 24th, 2017 14:53:40  
 Copyright © 2017, The President & Fellows of Harvard College

← **WorldMap layers available for the selected Geospatial Data Type: "Boston, Administrative Geography"**

**5. Submit the data for mapping!**

Map Data File

Select the type of geospatial data contained in your data file, and the column(s) in which it can be found.

**Geospatial Data**      **Type**      **Column Name**

Boston, Administrative Geography      public\_works\_id

Select the WorldMap layer to map your geospatial data onto.

**WorldMap Layer**      **Name**      **Description**

Submit information and wait for it to process.      2015 - Public Works Districts, Boston      Boston, Administrative Geography, Boston Public Works District ID. Examples: "2-06", "1-09", "1-1A", "1-1B"

Working... (This may take up to 45 seconds)      Cancel

**6. View Results**

At this point you will be presented with a basic map that can be styled to your specifications. The example pictured below includes an error message - some of the rows weren't able to be matched properly. In this case, you could still

go forward with your map, but without the information from the unmatched rows.

The screenshot shows the WorldMap interface. At the top, a green success message states "Success! — Map added to WorldMap. View on WorldMap." Below it, a yellow warning message reads: "Warning! — There were some issues. The map below was created from your tabular file." The warning lists: "Join Column: public\_works\_id", "Rows Matched: 8", and "Rows NOT Matched: 5 (View Map Metadata to see the unmatched rows)". A red arrow points from this warning to a text box that says "Any errors are displayed here, but the map can still be styled and used." Below the warning is a grey bar with the text: "Save this map and return to Dataverse. You can come back and make further edits to the map later if you like." This bar contains two buttons: "Return to the Dataverse" and "View Map Metadata". To the right of these buttons are three more buttons: "View on WorldMap", "Download", and "Delete". Below this is the "Style Map" section, which includes a map of Boston with several red-shaded areas. To the right of the map is a styling panel with fields for "Attribute", "Classification Method", "Intervals" (set to 5), and "Colors". Below the styling panel is an "Apply Changes" button and a legend showing a red square. A red arrow points from the text "Use these options to style your map." to the styling panel.

## Finalizing your map

Now that you have created your map:

- It exists on the WorldMap platform and may be viewed there – with all of WorldMap’s capabilities.
- Dataverse will contain a preview of the map and links to the larger version on WorldMap.

The map editor (pictured above) provides a set of options you can use to style your map. Clicking “Apply Changes” saves the current version of your map to Dataverse and Worldmap. The “Return to the Dataverse” button brings you back to Dataverse. “View on WorldMap” takes you to the map’s page on WorldMap, which offers additional views and options.

If you’d like to make further changes to your map in the future, you can return to the editor by clicking the “Map Data” button on your file.

## Removing your map

You can delete your map at any time. If you are on Dataverse, click “Map Data” and click the “Delete Map” button on the upper right. This completely removes the map and underlying data from the WorldMap platform.

## 1.7 Appendix

Additional documentation complementary to the User Guide.

### Contents:

- *Metadata References*

### 1.7.1 Metadata References

Dataverse is committed to using standard-compliant metadata to ensure that Dataverse metadata can be mapped easily to standard metadata schemas and be exported into JSON format (XML for tabular file metadata) for preservation and interoperability.

Detailed below are what metadata schemas we support for Citation and Domain Specific Metadata in Dataverse:

- **Citation Metadata:** compliant with [DDI Lite](#), [DDI 2.5 Codebook](#), [DataCite 3.1](#), and [Dublin Core's DCMI Metadata Terms](#) (see [.tsv version](#)). Language field uses [ISO 639-1](#) controlled vocabulary.
- **Geospatial Metadata:** compliant with [DDI Lite](#), [DDI 2.5 Codebook](#), [DataCite](#), and [Dublin Core](#) (see [.tsv version](#)). Country / Nation field uses [ISO 3166-1](#) controlled vocabulary.
- **Social Science & Humanities Metadata:** compliant with [DDI Lite](#), [DDI 2.5 Codebook](#), and [Dublin Core](#) (see [.tsv version](#)).
- **Astronomy and Astrophysics Metadata :** These metadata elements can be mapped/exported to the [International Virtual Observatory Alliance's \(IVOA\) VOResource Schema](#) format and is based on [Virtual Observatory \(VO\) Discovery and Provenance Metadata](#) (see [.tsv version](#)).
- **Life Sciences Metadata:** based on [ISA-Tab Specification](#), along with controlled vocabulary from subsets of the [OBI Ontology](#) and the [NCBI Taxonomy for Organisms](#) (see [.tsv version](#)).
- **Journal Metadata:** based on the [Journal Archiving and Interchange Tag Set, version 1.2](#) (see [.tsv version](#)).

See also the [Dataverse 4.0 Metadata Crosswalk: DDI, DataCite, DC, DCTerms, VO, ISA-Tab](#) document and the [Metadata Customization](#) section of the Admin Guide.

## ADMIN GUIDE

This guide documents the functionality only available to superusers (such as “dataverseAdmin”) through the *Dashboard* as well as command line utilities of interest to sysadmins.

### Contents:

## 2.1 Dashboard

Dataverse offers a dashboard of administrative tools for superusers only. If you are a logged-in superuser, you can access it by clicking your username in the navbar, and then clicking “Dashboard” from the dropdown. You can verify that you are a superuser by checking the color of your username in the navbar. If it’s red, you have the right permissions to use the Dashboard. Superusers can give other users the superuser status via *User Administration*.

### Contents:

- *Harvesting*
  - *Harvesting Clients*
  - *Harvesting Servers*
- *Metadata Export*
- *Users*
- *Move Data*

### 2.1.1 Harvesting

#### Harvesting Clients

This dashboard tool allows you to set up which other repositories your Dataverse installation harvests metadata records from. You can see a list of harvesting clients and add, edit, or remove them. See the *Managing Harvesting Clients* section for more details.

#### Harvesting Servers

This dashboard tool allows you to define sets of local datasets to make available to remote harvesting clients. You can see a list of sets and add, edit, or remove them. See the *Managing Harvesting Server and Sets* section for more details.

## 2.1.2 Metadata Export

This part of the Dashboard is simply a reminder message that metadata export happens through the Harvard Dataverse API. See the *Metadata Export* section and the *Native API* section of the API Guide for more details.

## 2.1.3 Users

This dashboard tool allows you to search a list of all users of your Dataverse installation. You can remove roles from user accounts and assign or remove superuser status. See the *User Administration* section for more details.

## 2.1.4 Move Data

This tool allows you to move datasets. To move dataverses, see the *Managing Datasets and Dataverses* section.

## 2.2 External Tools

External tools can provide additional features that are not part of Dataverse itself, such as data exploration.

### Contents:

- *Inventory of External Tools*
- *Managing External Tools*
  - *Adding External Tools to Dataverse*
  - *Listing All External Tools in Dataverse*
  - *Showing an External Tool in Dataverse*
  - *Removing an External Tool From Dataverse*
- *Testing External Tools*
  - *File Level Explore Tools*
  - *File Level Configure Tools*
  - *Dataset Level Explore Tools*
  - *Dataset Level Configure Tools*
- *Writing Your Own External Tool*



## 2.2.1 Inventory of External Tools

Tool	Type	Scope	Description
TwoRavens	ex- plore	file	A system of interlocking statistical tools for data exploration, analysis, and meta-analysis: <a href="http://2ra.vn">http://2ra.vn</a> . See the <i>TwoRavens: Tabular Data Exploration</i> section of the User Guide for more information on TwoRavens from the user perspective and the <i>TwoRavens</i> section of the Installation Guide.
Data Explorer	ex- plore	file	A GUI which lists the variables in a tabular data file allowing searching, charting and cross tabulation analysis. See the README.md file at <a href="https://github.com/scholarsportal/Dataverse-Data-Explorer">https://github.com/scholarsportal/Dataverse-Data-Explorer</a> for the instructions on adding Data Explorer to your Dataverse; and the <i>Prerequisites</i> section of the Installation Guide for the instructions on how to set up <b>basic R configuration required</b> (specifically, Dataverse uses R to generate .prep metadata files that are needed to run Data Explorer).
Whole Tale	ex- plore	dataset	A platform for the creation of reproducible research packages that allows users to launch containerized interactive analysis environments based on popular tools such as Jupyter and RStudio. Using this integration, Dataverse users can launch Jupyter and RStudio environments to analyze published datasets. For more information, see the <a href="#">Whole Tale User Guide</a> .
File Pre-viewers	ex- plore	file	A set of tools that display the content of files - including audio, html, <a href="#">Hypothes.is</a> annotations, images, PDF, text, video - allowing them to be viewed without downloading. The previewers can be run directly from github.io, so the only required step is using the Dataverse API to register the ones you want to use. Documentation, including how to optionally brand the previewers, and an invitation to contribute through github are in the README.md file. <a href="https://github.com/QualitativeDataRepository/dataverse-previewers">https://github.com/QualitativeDataRepository/dataverse-previewers</a>
Data Curation Tool	con- fig- ure	file	A GUI for curating data by adding labels, groups, weights and other details to assist with informed reuse. See the README.md file at <a href="https://github.com/scholarsportal/Dataverse-Data-Curation-Tool">https://github.com/scholarsportal/Dataverse-Data-Curation-Tool</a> for the installation instructions.

## 2.2.2 Managing External Tools

### Adding External Tools to Dataverse

To add an external tool to your installation of Dataverse you must first download a JSON file for that tool, which we refer to as a “manifest”. It should look something like this:

```
{
  "displayName": "Fabulous File Tool",
  "description": "Fabulous Fun for Files!",
  "scope": "file",
  "type": "explore",
  "toolUrl": "https://fabulousfiletool.com",
  "contentType": "text/tab-separated-values",
  "toolParameters": {
    "queryParameters": [
      {
        "fileid": "{fileId}"
      },
      {
        "key": "{apiToken}"
      }
    ]
  }
}
```

```
}  
}
```

Go to [Inventory of External Tools](#) and download a JSON manifest for one of the tools by following links in the description to installation instructions.

In the curl command below, replace the placeholder “fabulousFileTool.json” placeholder for the actual name of the JSON file you downloaded.

```
curl -X POST -H 'Content-type: application/json' http://localhost:8080/api/admin/  
externalTools --upload-file fabulousFileTool.json
```

### Listing All External Tools in Dataverse

To list all the external tools that are available in Dataverse:

```
curl http://localhost:8080/api/admin/externalTools
```

### Showing an External Tool in Dataverse

To show one of the external tools that are available in Dataverse, pass its database id:

```
export TOOL_ID=1  
curl http://localhost:8080/api/admin/externalTools/$TOOL_ID
```

### Removing an External Tool From Dataverse

Assuming the external tool database id is “1”, remove it with the following command:

```
export TOOL_ID=1  
curl -X DELETE http://localhost:8080/api/admin/externalTools/$TOOL_ID
```

## 2.2.3 Testing External Tools

Once you have added an external tool to your installation of Dataverse, you will probably want to test it to make sure it is functioning properly.

### File Level Explore Tools

File level explore tools are specific to the file type (content type or MIME type) of the file. For example, there is a tool for exploring PDF files in the “File Previewers” set of tools.

An “Explore” button will appear (on both the dataset page and the file landing page) for files that match the type that the tool has been built for. When there are multiple explore tools for a filetype, the button becomes a dropdown.

### File Level Configure Tools

File level configure tools are only available when you log in and have write access to the file. The file type determines if a configure tool is available. For example, a configure tool may only be available for tabular files.

## Dataset Level Explore Tools

When a dataset level explore tool is added, an “Explore” button on the dataset page will appear. This button becomes a drop down when there are multiple tools.

## Dataset Level Configure Tools

Configure tools at the dataset level are not currently supported. No button appears in the GUI if you add this type of tool.

## 2.2.4 Writing Your Own External Tool

If you plan to write a external tool, see the *Building External Tools* section of the API Guide.

If you have an idea for an external tool, please let the Dataverse community know by posting about it on the dataverse-community mailing list: <https://groups.google.com/forum/#!forum/dataverse-community>

## 2.3 Managing Harvesting Clients

### Contents:

- *Your Dataverse as a Metadata Harvester*
- *Managing Harvesting Clients*
- *New in Dataverse 4, vs. DVN 3*

### 2.3.1 Your Dataverse as a Metadata Harvester

Harvesting is a process of exchanging metadata with other repositories. As a harvesting *client*, your Dataverse can gather metadata records from remote sources. These can be other Dataverse instances or other archives that support OAI-PMH, the standard harvesting protocol. Harvested metadata records will be indexed and made searchable by your users. Clicking on a harvested dataset in the search results takes the user to the original repository. Harvested datasets cannot be edited in your Dataverse installation.

Harvested records can be kept in sync with the original repository through scheduled incremental updates, daily or weekly. Alternatively, harvests can be run on demand, by the Admin.

### 2.3.2 Managing Harvesting Clients

To start harvesting metadata from a remote OAI repository, you first create and configure a *Harvesting Client*.

Clients are managed on the “Harvesting Clients” page accessible via the *Dashboard*. Click on the *Add Client* button to get started.

The process of creating a new, or editing an existing client, is largely self-explanatory. It is split into logical steps, in a way that allows the user to go back and correct the entries made earlier. The process is interactive and guidance text is provided. For example, the user is required to enter the URL of the remote OAI server. When they click *Next*, the application will try to establish a connection to the server in order to verify that it is working, and to obtain the information about the sets of metadata records and the metadata formats it supports. The choices offered to the user

on the next page will be based on this extra information. If the application fails to establish a connection to the remote archive at the address specified, or if an invalid response is received, the user is given an opportunity to check and correct the URL they entered.

### 2.3.3 New in Dataverse 4, vs. DVN 3

- Note that when creating a client you will need to select an existing local dataverse to host the datasets harvested. In DVN 3, a dedicated “harvesting dataverse” would be created specifically for each remote harvesting source. In Dataverse 4, harvested content can be added to *any dataverse*. This means that a dataverse can now contain datasets harvested from multiple sources and/or a mix of local and harvested datasets.
- An extra “Archive Type” pull down menu is added to the Create and Edit dialogs. This setting, selected from the choices such as “Dataverse 4”, “DVN, v2-3”, “Generic OAI”, etc. is used to properly format the harvested metadata as they are shown in the search results. It is **very important** to select the type that best describes this remote server, as failure to do so can result in information missing from the search results, and, a **failure to redirect the user to the archival source** of the data!

It is, however, **very easy to correct** a mistake like this. For example, let’s say you have created a client to harvest from the XYZ Institute and specified the archive type as “Dataverse 4”. You have been able to harvest content, the datasets appear in search result, but clicking on them results in a “Page Not Found” error on the remote site. At which point you realize that the XYZ Institute admins have not yet upgraded to Dataverse 4, still running DVN v3.1.2 instead. All you need to do is go back to the Harvesting Clients page, and change the setting to “DVN, v2-3”. This will fix the redirects **without having to re-harvest** the datasets.

- Another extra entry, “Archive Description”, is added to the *Edit Harvesting Client* dialog. This description appears at the bottom of each search result card for a harvested dataset or datafile. By default, this text reads “This Dataset is harvested from our partners. Clicking the link will take you directly to the archival source of the data.” Here it can be customized to be more descriptive, for example, “This Dataset is harvested from our partners at the XYZ Institute...”

## 2.4 Managing Harvesting Server and Sets

### Contents:

- *Your Dataverse as an OAI server*
- *How does it work?*
- *OAI Sets*
  - *Important: New SOLR schema required!*
- *OAI Set updates*

### 2.4.1 Your Dataverse as an OAI server

As a harvesting *server*, your Dataverse can make some of the local dataset metadata available to remote harvesting clients. These can be other Dataverse instances, or any other clients that support OAI-PMH harvesting protocol. Note that the terms “Harvesting Server” and “OAI Server” are being used interchangeably throughout this guide and in the inline help text.

If you want to learn more about OAI-PMH, you could take a look at [DataCite OAI-PMH guide](#) or the [OAI-PMH protocol definition](#).

You might consider adding your OAI-enabled production instance of Dataverse to [this shared list](#) of such instances.

## 2.4.2 How does it work?

Only the published, unrestricted datasets in your Dataverse can be made harvestable. Remote clients normally keep their records in sync through scheduled incremental updates, daily or weekly, thus minimizing the load on your server. Note that it is only the metadata that are harvested. Remote harvesters will generally not attempt to download the data files associated with the harvested datasets.

Harvesting server can be enabled or disabled on the “Harvesting Server” page accessible via the *Dashboard*. Harvesting server is by default disabled on a brand new, “out of the box” Dataverse.

The OAI-PMH endpoint can be accessed at `http(s)://<Your Dataverse FQDN>/oai`. If you want other services to harvest your repository, point them to this URL. *Example URL for ‘Identify’ verb:* `demo.dataverse.org OAI`

## 2.4.3 OAI Sets

Once the service is enabled, you define collections of local datasets that will be available to remote harvesters as *OAI Sets*. Once again, the terms “OAI Set” and “Harvesting Set” are used interchangeably. Sets are defined by search queries. Any such query that finds any number of published, local (non-harvested) datasets can be used to create an OAI set. Sets can overlap local dataverses, and can include as few or as many of your local datasets as you wish. A good way to master the Dataverse search query language is to experiment with the Advanced Search page. We also recommend that you consult the Search API section of the Dataverse User Guide.

Once you have entered the search query and clicked *Next*, the number of search results found will be shown on the next screen. This way, if you are seeing a number that’s different from what you expected, you can go back and try to re-define the query.

Some useful examples of search queries to define OAI sets:

- A good way to create a set that would include all your local, published datasets is to do so by the Unique Identifier authority registered to your Dataverse, for example:

```
dsPersistentId:"doi:1234/"
```

Note that double quotes must be used, since the search field value contains the colon symbol!

Note also that the search terms limiting the results to published and local datasets **are added to the query automatically**, so you don’t need to worry about that.

- A query to create a set to include the datasets from a specific local dataverse:

```
parentId:NNN
```

where NNN is the database id of the dataverse object (consult the Dataverse table of the SQL database used by the application to verify the database id).

- A query to find all the dataset by a certain author:

```
authorName:YYY
```

where YYY is the name.

- Complex queries can be created with multiple logical AND and OR operators. For example,

```
(authorName:YYY OR authorName:ZZZ) AND dsPublicationDate:NNNN
```

- Some further query examples:

For specific datasets using a persistentID:

```
(dsPersistentId:10.5000/ZZYYXX/ OR dsPersistentId:10.5000/XXYYZZ)
```

For all datasets within a specific ID authority:

```
dsPersistentId:10.5000/XXYYZZ
```

For all dataverses with subjects of Astronomy and Astrophysics or Earth and Environmental Sciences:

```
(dvSubject:"Astronomy and Astrophysics" OR dvSubject:"Earth and  
Environmental Sciences")
```

For all datasets containing the keyword “censorship”:

```
keywordValue:censorship
```

### Important: New SOLR schema required!

In order to be able to define OAI sets, your SOLR server must be upgraded with the search schema that came with the Dataverse release 4.5 (or later), and all your local datasets must be re-indexed, once the new schema is installed.

## 2.4.4 OAI Set updates

Every time a new harvesting set is created, or changes are made to an existing set, the contents of the set are automatically updated - the Dataverse application will find the datasets defined by the query, and attempt to run the metadata export on the ones that haven’t been exported yet. Only the datasets for which the export has completed successfully, and the results cached on the filesystem are included in the OAI sets advertised to the harvesting clients!

This is in contrast to how the sets used to be managed in DVN v.3, where sets had to be exported manually before any such changes had effect.

**Important:** Note however that changes made to the actual dataset metadata do not automatically trigger any corresponding OAI sets to be updated immediately! For example: let’s say you have created an OAI set defined by the search query `authorName:king`, that resulted in 43 dataset records. If a new dataset by the same author is added and published, this **does not** immediately add the extra record to the set! It would simply be too expensive, to refresh all the sets every time any changes to the metadata are made.

The OAI set will however be updated automatically by a scheduled metadata export job that runs every night (at 2AM, by default). This export timer is created and activated automatically every time the application is deployed or restarted. Once again, this is new in Dataverse 4, and unlike DVN v3, where export jobs had to be scheduled and activated by the admin user. See the “Export” section of the Admin guide, for more information on the automated metadata exports.

It is still possible however to make changes like this be immediately reflected in the OAI server, by going to the *Harvesting Server* page and clicking the “Run Export” icon next to the desired OAI set.

## 2.5 Metadata Customization

Dataverse has a flexible data-driven metadata system powered by “metadata blocks” that are listed in the *Appendix* section of the User Guide. In this section we explain the customization options.

### Contents:

- *Introduction*
- *About the metadata block TSV*
  - *#metadataBlock properties*

- *#datasetField (field) properties*
- *#controlledVocabulary (enumerated) properties*
- *FieldType definitions*
- *displayFormat variables*
- *Metadata Block Setup*
  - *Exploring Metadata Blocks*
  - *Setting Up a Dev Environment for Testing*
  - *Editing TSV files*
  - *Loading TSV files into Dataverse*
  - *#metadataBlock properties*
  - *#datasetField (field) properties*
  - *#controlledVocabulary (enumerated) properties*
  - *Enabling a Metadata Block*
  - *Updating the Solr Schema*
- *Reloading a Metadata Block*
- *Tips from the Dataverse Community*
- *Footnotes*

## 2.5.1 Introduction

Before you embark on customizing metadata in Dataverse you should make sure you are aware of the modest amount of customization that is available with the Dataverse web interface. It's possible to hide fields and make field required by clicking "Edit" at the dataverse level, clicking "General Information" and making adjustments under "Metadata Fields" as described in the context of dataset templates in the *Dataverse Management* section of the User Guide.

Much more customization of metadata is possible, but this is an advanced topic so feedback on what is written below is very welcome. The possibilities for customization include:

- Editing and adding metadata fields
- Editing and adding instructional text (field label tooltips and text box watermarks)
- Editing and adding controlled vocabularies
- Changing which fields depositors must use in order to save datasets (see also "dataset templates" in the *Dataverse Management* section of the User Guide.)
- Changing how saved metadata values are displayed in the UI

Generally speaking it is safer to create your own custom metadata block rather than editing metadata blocks that ship with Dataverse, because changes to these blocks may be made in future releases of Dataverse. If you'd like to make improvements to any of the metadata blocks shipped with Dataverse, please open an issue at <https://github.com/IQSS/dataverse/issues> so it can be discussed before a pull request is made. Please note that the metadata blocks shipped with Dataverse are based on standards (e.g. DDI for social science) and you can learn more about these standards in the *Appendix* section of the User Guide. If you have developed your own custom metadata block that you think may be of interest to the Dataverse community, please create an issue and consider making a pull request as described in the *Version Control* section of the Developer Guide.

In Dataverse 4, custom metadata are no longer defined as individual fields, as they were in Dataverse Network (DVN) 3.x, but in metadata blocks. Dataverse 4 ships with a citation metadata block, which includes mandatory fields needed for assigning persistent IDs to datasets, and domain specific metadata blocks. For a complete list, see the *Appendix* section of the User Guide.

Definitions of these blocks are loaded into a Dataverse installation in tab-separated value (TSV).<sup>1,2</sup> While it is technically possible to define more than one metadata block in a TSV file, it is good organizational practice to define only one in each file.

The metadata block TSVs shipped with Dataverse are in [this folder in the Dataverse github repo](#) and the corresponding ResourceBundle property files are [here](#). Human-readable copies are available in [this Google Sheets document](#) but they tend to get out of sync with the TSV files, which should be considered authoritative. The Dataverse installation process operates on the TSVs, not the Google spreadsheet.

### 2.5.2 About the metadata block TSV

Here we list and describe the purposes of each section and property of the metadata block TSV.

#### 1. metadataBlock

- Purpose: Represents the metadata block being defined.
- Cardinality:
  - 0 or more per Dataverse
  - 1 per Metadata Block definition

#### 2. datasetField

- Purpose: Each entry represents a metadata field to be defined within a metadata block.
- Cardinality: 1 or more per metadataBlock

#### 3. controlledVocabulary

- Purpose: Each entry enumerates an allowed value for a given datasetField.
- Cardinality: zero or more per datasetField

Each of the three main sections own sets of properties:

---

<sup>1</sup> <https://www.iana.org/assignments/media-types/text/tab-separated-values>

<sup>2</sup> Although the structure of the data, as you'll see below, violates the "Each record must have the same number of fields" tenet of TSV



**#metadataBlock properties**

Property	Purpose	Allowed values and restrictions
name	A user-definable string used to identify a #metadataBlock	<ul style="list-style-type: none"> <li>• No spaces or punctuation, except underscore.</li> <li>• By convention, should start with a letter, and use lower camel case<sup>3</sup></li> <li>• Must not collide with a field of the same name in the same or any other #datasetField definition, including metadata blocks defined elsewhere.<sup>4</sup></li> </ul>
dataverseAlias	If specified, this metadata block will be available only to the dataverse designated here by its alias and to children of that dataverse.	Free text. For an example, see custom_hbgdki.tsv.
displayName	Acts as a brief label for display related to this #metadataBlock.	Should be relatively brief. The limit is 256 character, but very long names might cause display problems.

<sup>3</sup> <https://en.wikipedia.org/wiki/CamelCase>

<sup>4</sup> These field names are added to the Solr schema.xml and cannot be duplicated. See "Editing TSV files" for how to check for duplication.

#datasetField (field) properties

Property	Purpose	Allowed values and restrictions
name	A user-definable string used to identify a #datasetField. Maps directly to field name used by Solr.	<ul style="list-style-type: none"> <li>• (from DatasetFieldType.java) The internal DDI-like name, no spaces, etc.</li> <li>• (from Solr) Field names should consist of alphanumeric or underscore characters only and not start with a digit. This is not currently strictly enforced, but other field names will not have first class support from all components and back compatibility is not guaranteed. Names with both leading and trailing underscores (e.g. _version_) are reserved.</li> <li>• Must not collide with a field of the same name in another #metadataBlock definition or any name already included as a field in the Solr index.</li> </ul>
title	Acts as a brief label for display related to this #datasetField.	Should be relatively brief.
description	Used to provide a description of the field.	Free text
watermark	A string to initially display in a field as a prompt for what the user should enter.	Free text
fieldType	Defines the type of content that the field, if not empty, is meant to contain.	<ul style="list-style-type: none"> <li>• none</li> <li>• date</li> <li>• email</li> <li>• text</li> <li>• textbox</li> <li>• url</li> <li>• int</li> <li>• float</li> <li>• See below for fieldtype definitions</li> </ul>
displayOrder	Controls the sequence in which the fields are displayed, both for input and presentation.	Non-negative integer.
displayFormat	Controls how the content is displayed for presentation (not entry). The value of this field may contain one or more special variables (enumerated below). HTML tags, likely in conjunction with one or more of these values, may be used to control the display of content in the web UI.	See below for displayFormat variables
advancedSearchField	Specify whether this field is available in advanced search.	TRUE (available) or FALSE (not available)
allowControlledVocabulary	Specify whether the possible values of this field are determined by values in the #controlledVocabulary section.	TRUE (controlled) or FALSE (not controlled)

**#controlledVocabulary (enumerated) properties**

Property	Purpose	Allowed values and restrictions
DatasetField	Specifies the #datasetField to which this entry applies.	Must reference an existing #datasetField. As a best practice, the value should reference a #datasetField in the current metadata block definition. (It is technically possible to reference an existing #datasetField from another metadata block.)
Value	A short display string, representing an enumerated value for this field. If the identifier property is empty, this value is used as the identifier.	Free text
identifier	A string used to encode the selected enumerated value of a field. If this property is empty, the value of the “Value” field is used as the identifier.	Free text
displayOrder	Control the order in which the enumerated values are displayed for selection.	Non-negative integer.

**FieldType definitions**

Field-type	Definition
none	Used for compound fields, in which case the parent field would have no value and display no data entry control.
date	A date, expressed in one of three resolutions of the form YYYY-MM-DD, YYYY-MM, or YYYY.
email	A valid email address. Not indexed for privacy reasons.
text	Any text other than newlines may be entered into this field.
textbox	Any text may be entered. For input, Dataverse presents a multi-line area that accepts newlines. While any HTML is permitted, only a subset of HTML tags will be rendered in the UI. <a href="#">A list of supported tags is included in the Dataverse User Guide</a> .
url	If not empty, field must contain a valid URL.
int	An integer value destined for a numeric field.
float	A floating point number destined for a numeric field.

**displayFormat variables**

These are common ways to use the displayFormat to control how values are displayed in the UI. This list is not exhaustive.

Variable	Description
(blank)	The displayFormat is left blank for primitive fields (e.g. subtitle) and fields that do not take values (e.g. author), since displayFormats do not work for these fields.
#VALUE	The value of the field (instance level).
#NAME	The name of the field (class level).
#EMAIL	For displaying emails.
<a href="#VALUE">#VALUE</a>	For displaying the value as a link (if the value entered is a link).
<a href='URL/#VALUE'>#VALUE</a>	For displaying the value as a link, with the value included in the URL (e.g. if URL is <a href="http://emsearch.rutgers.edu/atlas/#VALUE_summary.html">http://emsearch.rutgers.edu/atlas/#VALUE_summary.html</a> , and the value entered is 1001, the field is displayed as <a href="http://emsearch.rutgers.edu/atlas/1001_summary.html">1001</a> (hyperlinked to <a href="http://emsearch.rutgers.edu/atlas/1001_summary.html">http://emsearch.rutgers.edu/atlas/1001_summary.html</a> )).
 	For displaying the image of an entered image URL (used to display images in the producer and distributor logos metadata fields).
#VALUE: - #VALUE: (#VALUE)	Appends and/or prepends characters to the value of the field. e.g. if the displayFormat for the distributorAffiliation is (#VALUE) (wrapped with parens) and the value entered is University of North Carolina, the field is displayed in the UI as (University of North Carolina).
; : ,	Displays the character (e.g. semicolon, comma) between the values of fields within compound fields. For example, if the displayFormat for the compound field "series" is a colon, and if the value entered for seriesName is IMPs and for seriesInformation is A collection of NMR data, the compound field is displayed in the UI as IMPs: A collection of NMR data.

### 2.5.3 Metadata Block Setup

Now that you understand the TSV format used for metadata blocks, the next step is to attempt to make improvements to existing metadata blocks or create entirely new metadata blocks. For either task, you should have a Dataverse environment set up for testing where you can drop the database frequently while you make edits to TSV files. Once you have tested your TSV files, you should consider making a pull request to contribute your improvement back to the community.

#### Exploring Metadata Blocks

In addition to studying the TSV files themselves you might find the following highly experimental and subject-to-change API endpoints useful to understand the metadata blocks that have already been loaded into your installation of Dataverse:

You can get a dump of metadata fields (yes, the output is odd, please open a issue) like this:

```
curl http://localhost:8080/api/admin/datasetfield
```

To see details about an individual field such as "title" in the example below:

```
curl http://localhost:8080/api/admin/datasetfield/title
```

## Setting Up a Dev Environment for Testing

You have several options for setting up a dev environment for testing metadata block changes:

- Vagrant: See the *Tools* section of the Dev Guide.
- docker-aiio: See <https://github.com/IQSS/dataverse/tree/develop/conf/docker-aiio>
- AWS deployment: See the *Deployment* section of the Dev Guide.
- Full dev environment: See the *Development Environment* section of the Dev Guide.

To get a clean environment in Vagrant, you'll be running `vagrant destroy`. In Docker, you'll use `docker rm`. For a full dev environment or AWS installation, you might find `rebuild` and related scripts at `scripts/deploy/` `phoenix.dataverse.org` useful.

## Editing TSV files

Early in Dataverse 4 development metadata blocks were edited in the Google spreadsheet mentioned above and then exported in TSV format. This worked fine when there was only one person editing the Google spreadsheet but now that contributions are coming in from all over, the TSV files are edited directly. We are somewhat painfully aware that another format such as XML might make more sense these days. Please see <https://github.com/IQSS/dataverse/issues/4451> for a discussion of non-TSV formats.

Please note that metadata fields share a common namespace so they must be unique. The following curl command will print list of metadata fields already available in the system:

```
curl http://localhost:8080/api/admin/index/solr/schema
```

We'll use this command again below to update the Solr schema to accomodate metadata fields we've added.

## Loading TSV files into Dataverse

A number of TSV files are loaded into Dataverse on every new installation, becoming the metadata blocks you see in the UI. For the list of metadata blocks that are included with Dataverse out of the box, see the *Appendix* section of the User Guide.

Along with TSV file, there are corresponding ResourceBundle property files with key=value pair [here](#). To add other language files, see the *Configuration* for `dataverse.lang.directory` JVM Options section, and add a file, for example: "citation\_lang.properties" to the path you specified for the `dataverse.lang.directory` JVM option, and then restart Glassfish.

If you are improving an existing metadata block, the Dataverse installation process will load the TSV for you, assuming you edited the TSV file in place. The TSV file for the Citation metadata block, for example, can be found at `scripts/api/data/metadatablocks/citation.tsv`. If any of the below mentioned property values are changed, corresponding ResourceBundle property file has to be edited and stored under `dataverse.lang.directory` location

- name, displayName property under #metadataBlock
- name, title, description, watermark properties under #datasetfield
- DatasetField, Value property under #controlledVocabulary

If you are creating a new custom metadata block (hopefully with the idea of contributing it back to the community if you feel like it would provide value to others), the Dataverse installation process won't know about your new TSV file so you must load it manually. The script that loads the TSV files into the system is `scripts/api/setup-datasetfields.sh` and contains a series of curl commands. Here's an example of the necessary curl command with the new custom metadata block in the "/tmp" directory.

```
curl http://localhost:8080/api/admin/datasetfield/load -H "Content-type:
text/tab-separated-values" -X POST --upload-file /tmp/new-metadata-block.tsv
```

To create a new ResourceBundle, here are the steps to generate key=value pair for the three main sections:

### #metadataBlock properties

metadatablock.name=(the value of **name** property from #metadatablock)

metadatablock.displayName=(the value of **displayName** property from #metadatablock)

example:

metadatablock.name=citation

metadatablock.displayName=Citation Metadata

### #datasetField (field) properties

datasetfieldtype.(the value of **name** property from #datasetField).title=(the value of **title** property from #datasetField)

datasetfieldtype.(the value of **name** property from #datasetField).description=(the value of **description** property from #datasetField)

datasetfieldtype.(the value of **name** property from #datasetField).watermark=(the value of **watermark** property from #datasetField)

example:

datasetfieldtype.title.title=Title

datasetfieldtype.title.description=Full title by which the Dataset is known.

datasetfieldtype.title.watermark=Enter title...

### #controlledVocabulary (enumerated) properties

controlledvocabulary.(the value of **DatasetField** property from #controlledVocabulary).(the value of **Value** property from #controlledVocabulary)=(the value of **Value** property from #controlledVocabulary)

Since the **Value** property from #controlledVocabulary is free text, while creating the key, it has to be converted to lowercase, replace space with underscore, and strip accents.

example:

controlledvocabulary.subject.agricultural\_sciences=Agricultural Sciences

controlledvocabulary.language.marathi\_(marathi)=Marathi (Maru0101u1E6Dhu012B)

### Enabling a Metadata Block

Running a curl command like “load” example above should make the new custom metadata block available within the system but in order to start using the fields you must either enable it from the GUI (see “General Information” in the *Dataverse Management* section of the User Guide) or by running a curl command like the one below using a superuser API token. In the example below we are enabling the “journal” and “geospatial” metadata blocks for the root dataverse:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X POST -H "Content-type:application/json" -d "[\"journal\", \"geospatial\"]" http://localhost:8080/api/dataverses/:root/metadatablocks
```

## Updating the Solr Schema

Once you have enabled a new metadata block you should be able to see the new fields in the GUI but before you can save the dataset, you must add additional fields to your Solr schema.

An API endpoint of Dataverse provides you with a generated set of all fields that need to be added to the Solr schema configuration, including any enabled metadata schemas:

```
curl http://localhost:8080/api/admin/index/solr/schema
```

For convenience and automation you can download and consider running `updateSchemaMDB.sh`. It uses the API endpoint above and writes schema files to the filesystem (so be sure to run it on the Solr server itself as the Unix user who owns the Solr files) and then triggers a Solr reload.

By default, it will download from Dataverse at `http://localhost:8080` and reload Solr at `http://localhost:8983`. You may use the following environment variables with this script or mix'n'match with options:

Environment variable	Option	Description	Example
<code>DATA-VERSE_URL</code>	<code>-d</code>	Provide the URL to your Dataverse installation	<code>http://localhost:8080</code>
<code>SOLR_URL</code>	<code>-s</code>	Provide the URL to your Solr instance	<code>http://localhost:8983</code>
<code>UN-BLOCK_KEY</code>	<code>-u</code>	If your installation has a blocked admin API endpoint, you can provide either the key itself or a path to a keyfile	<code>xyz</code> or <code>/secrets/unblock.key</code>
<code>TARGET</code>	<code>-t</code>	Provide the config directory of your Solr core “collection1”	<code>/usr/local/solr/solr-7.3.1/server/solr/collection1/conf</code>

See the *Prerequisites* section of the Installation Guide for a suggested location on disk for the Solr schema file.

Please note that if you are going to make a pull request updating `conf/solr/7.3.1/schema.xml` with fields you have added, you should first load all the custom metadata blocks in `scripts/api/data/metadatablocks` (including ones you don't care about) to create a complete list of fields.

## 2.5.4 Reloading a Metadata Block

As mentioned above, changes to metadata blocks that ship with Dataverse will be made over time to improve them and release notes will sometimes instruct you to reload an existing metadata block. The syntax for reloading is the same as reloading. Here's an example with the “citation” metadata block:

```
curl http://localhost:8080/api/admin/datasetfield/load -H "Content-type:text/tab-separated-values" -X POST --upload-file citation.tsv
```

Great care must be taken when reloading a metadata block. Matching is done on field names (or identifiers and then names in the case of controlled vocabulary values) so it's easy to accidentally create duplicate fields.

The ability to reload metadata blocks means that SQL update scripts don't need to be written for these changes. See also the *SQL Upgrade Scripts* section of the Dev Guide.

## 2.5.5 Tips from the Dataverse Community

If there are tips that you feel are omitted from this document, please open an issue at <https://github.com/IQSS/dataverse/issues> and consider making a pull request to make improvements. You can find this document at <https://github.com/IQSS/dataverse/issues>

[//github.com/IQSS/dataverse/blob/develop/doc/sphinx-guides/source/admin/metadatacustomization.rst](https://github.com/IQSS/dataverse/blob/develop/doc/sphinx-guides/source/admin/metadatacustomization.rst)

Alternatively, you are welcome to request “edit” access to this “Tips for Dataverse metadata blocks from the community” Google doc: <https://docs.google.com/document/d/1XpblRw0v0SvV-Bq6njlN96WyHJ7tqG0WWejqBdl7hE0/edit?usp=sharing>

The thinking is that the tips can become issues and the issues can eventually be worked on as features to improve the Dataverse metadata system.

## 2.5.6 Footnotes

## 2.6 Metadata Export

### Contents:

- *Automatic Exports*
- *Batch exports through the API*
- *Export Failures*
- *Downloading Metadata via GUI*
- *Downloading Metadata via API*

### 2.6.1 Automatic Exports

Publishing a dataset automatically starts a metadata export job, that will run in the background, asynchronously. Once completed, it will make the dataset metadata exported and cached in all the supported formats listed under *Supported Metadata Export Formats* in the *Dataset + File Management* section of the User Guide.

A scheduled timer job that runs nightly will attempt to export any published datasets that for whatever reason haven’t been exported yet. This timer is activated automatically on the deployment, or restart, of the application. So, again, no need to start or configure it manually. (See the “Application Timers” section of this guide for more information)

### 2.6.2 Batch exports through the API

In addition to the automated exports, a Dataverse admin can start a batch job through the API. The following 2 API calls are provided:

```
curl http://localhost:8080/api/admin/metadata/exportAll
```

```
curl http://localhost:8080/api/admin/metadata/reExportAll
```

The former will attempt to export all the published, local (non-harvested) datasets that haven’t been exported yet. The latter will *force* a re-export of every published, local dataset, regardless of whether it has already been exported or not.

Note, that creating, modifying, or re-exporting an OAI set will also attempt to export all the unexported datasets found in the set.



### 2.6.3 Export Failures

An export batch job, whether started via the API, or by the application timer, will leave a detailed log in your configured logs directory. This is the same location where your main Glassfish server.log is found. The name of the log file is `export_[timestamp].log` - for example, `export_2016-08-23T03-35-23.log`. The log will contain the numbers of datasets processed successfully and those for which metadata export failed, with some information on the failures detected. Please attach this log file if you need to contact Dataverse support about metadata export problems.

### 2.6.4 Downloading Metadata via GUI

The *Dataset + File Management* section of the User Guide explains how end users can download the metadata formats above from the Dataverse GUI.

### 2.6.5 Downloading Metadata via API

The *Native API* section of the API Guide explains how end users can download the metadata formats above via API.

## 2.7 Dataverse Application Timers

Dataverse uses timers to automatically run scheduled Harvest and Metadata export jobs.

#### Contents:

- *Dedicated timer server in a Dataverse server cluster*
- *Harvesting Timers*
- *Metadata Export Timer*
- *Known Issues*

### 2.7.1 Dedicated timer server in a Dataverse server cluster

When running a Dataverse cluster - i.e. multiple Dataverse application servers talking to the same database - **only one** of them must act as the *dedicated timer server*. This is to avoid starting conflicting batch jobs on multiple nodes at the same time.

This does not affect a single-server installation. So you can safely skip this section unless you are running a multi-server cluster.

The following JVM option instructs the application to act as the dedicated timer server:

```
-Ddataverse.timerServer=true
```

**IMPORTANT:** Note that this option is automatically set by the Dataverse installer script. That means that when **configuring a multi-server cluster**, it will be the responsibility of the installer to remove the option from the `domain.xml` of every node except the one intended to be the timer server. We also recommend that the following entry in the `domain.xml`: `<ejb-timer-service timer-datasource="jdbc/VDCNetDS">` is changed back to `<ejb-timer-service>` on all the non-timer server nodes. Similarly, this option is automatically set by the installer script. Changing it back to the default setting on a server that doesn't need to run the timer will prevent a potential race condition, where multiple servers try to get a lock on the timer database.

**Note** that for the timer to work, the version of the PostgreSQL JDBC driver your instance is using must match the version of your PostgreSQL database. See the ‘Timer not working’ section of the [Troubleshooting](#) guide.

## 2.7.2 Harvesting Timers

These timers are created when scheduled harvesting is enabled by a local admin user (via the “Manage Harvesting Clients” page).

In a multi-node cluster, all these timers will be created on the dedicated timer node (and not necessarily on the node where the harvesting clients were created and/or saved).

A timer will be automatically removed when a harvesting client with an active schedule is deleted, or if the schedule is turned off for an existing client.

## 2.7.3 Metadata Export Timer

This timer is created automatically whenever the application is deployed or restarted. There is no admin user-accessible configuration for this timer.

This timer runs a daily job that tries to export all the local, published datasets that haven’t been exported yet, in all supported metadata formats, and cache the results on the filesystem. (Note that normally an export will happen automatically whenever a dataset is published. This scheduled job is there to catch any datasets for which that export did not succeed, for one reason or another). Also, since this functionality has been added in version 4.5: if you are upgrading from a previous version, none of your datasets are exported yet. So the first time this job runs, it will attempt to export them all.

This daily job will also update all the harvestable OAI sets configured on your server, adding new and/or newly published datasets or marking deaccessioned datasets as “deleted” in the corresponding sets as needed.

This job is automatically scheduled to run at 2AM local time every night. If really necessary, it is possible (for an advanced user) to change that time by directly editing the EJB timer application table in the database.

## 2.7.4 Known Issues

We’ve received several reports of an intermittent issue where the application fails to deploy with the error message “EJB Timer Service is not available.” Please see the [Troubleshooting](#) section of this guide for a workaround.

## 2.8 Make Data Count

[Make Data Count](#) is a project to collect and standardize metrics on data use, especially views, downloads, and citations. Dataverse can integrate Make Data Count to collect and display usage metrics including counts of dataset views, file downloads, and dataset citations.

### Contents:

- [Introduction](#)
- [Architecture](#)
- [Limitations for Dataverse Installations Using Handles Rather Than DOIs](#)
- [Configuring Dataverse for Make Data Count Views and Downloads](#)

- *Enable Logging for Make Data Count*
- *Configure Counter Processor*
- *Populate Views and Downloads for the First Time*
- *Populate Views and Downloads Nightly*
- *Sending Usage Metrics to the DataCite Hub*
- *Configuring Dataverse for Make Data Count Citations*
- *Retrieving Make Data Count Metrics from the DataCite Hub*
- *Retrieving Make Data Count Metrics from Dataverse*

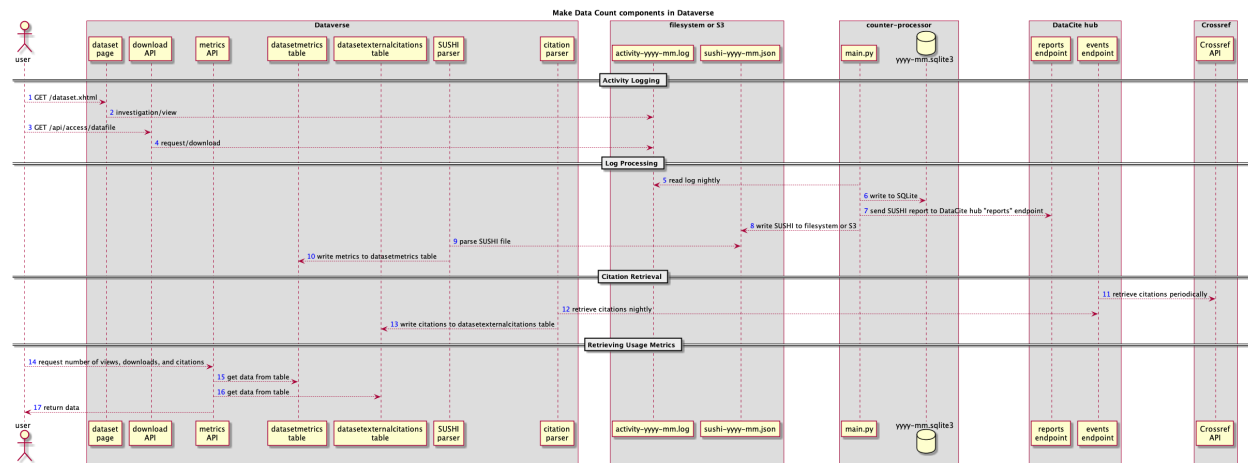
## 2.8.1 Introduction

Make Data Count is part of a broader Research Data Alliance (RDA) Data Usage Metrics Working Group which helped to produce a specification called the COUNTER Code of Practice for Research Data (PDF, HTML) that Dataverse makes every effort to comply with. The Code of Practice (CoP) is built on top of existing standards such as COUNTER and SUSHI that come out of the article publishing world. The Make Data Count project has emphasized that they would like feedback on the code of practice. You can keep up to date on the Make Data Count project by subscribing to their [newsletter](#).

## 2.8.2 Architecture

Dataverse installations who would like support for Make Data Count must install Counter Processor, a Python project created by California Digital Library (CDL) which is part of the Make Data Count project and which runs the software in production as part of their DASH data sharing platform.

The diagram below shows how Counter Processor interacts with Dataverse and the DataCite hub, once configured. Installations of Dataverse using Handles rather than DOIs should note the limitations in the next section of this page.



The most important takeaways from the diagram are:

- Once enabled, Dataverse will log activity (views and downloads) to a specialized date-stamped file.
- You should run Counter Processor once a day to create reports in SUSHI (JSON) format that are saved to disk for Dataverse to process and that are sent to the DataCite hub.

- You should set up a cron job to have Dataverse process the daily SUSHI reports, updating the Dataverse database with the latest metrics.
- You should set up a cron job to have Dataverse pull the latest list of citations for each dataset on a periodic basis, perhaps weekly or daily. These citations come from Crossref via the DataCite hub.
- APIs are available in Dataverse to retrieve Make Data Count metrics: views, downloads, and citations.

### 2.8.3 Limitations for Dataverse Installations Using Handles Rather Than DOIs

Data repositories using Handles and other identifiers are not supported by Make Data Count but in the [notes](#) following a July 2018 webinar, you can see the Make Data Count project’s response on this topic. In short, the DataCite hub does not want to receive reports for non-DOI datasets. Additionally, citations are only available from the DataCite hub for datasets that have DOIs. See also the table below.

	DOIs	Handles
<b>Out of the box</b>	Classic download counts	Classic download counts
<b>Make Data Count</b>	MDC views, MDC downloads, MDC citations	MDC views, MDC downloads

This being said, the Dataverse usage logging can still generate logs and process those logs with Counter Processor to create json that details usage on a dataset level. Dataverse can ingest this locally generated json.

When editing the `counter-processor-config.yaml` file mentioned below, make sure that the `upload_to_hub` boolean is set to `False`.

### 2.8.4 Configuring Dataverse for Make Data Count Views and Downloads

If you haven’t already, follow the steps for installing Counter Processor in the [Prerequisites](#) section of the Installation Guide.

#### Enable Logging for Make Data Count

To make Dataverse log dataset usage (views and downloads) for Make Data Count, you must set the `:MDCLogPath` database setting. See [:MDCLogPath](#) for details.

After you have your first day of logs, you can process them the next day.

#### Configure Counter Processor

- First, become the “counter” Unix user.
  - `sudo su - counter`
- Change to the directory where you installed Counter Processor.
  - `cd /usr/local/counter-processor-0.0.1`
- Download `counter-processor-config.yaml` to `/usr/local/counter-processor-0.0.1`.
- Edit the config file and pay particular attention to the `FIXME` lines.
  - `vim counter-processor-config.yaml`

## Populate Views and Downloads for the First Time

Soon we will be setting up a cron job to run nightly but we start with a single successful configuration and run of Counter Processor and calls to Dataverse APIs.

- Change to the directory where you installed Counter Processor.
  - `cd /usr/local/counter-processor-0.0.1`
- If you are running Counter Processor for the first time in the middle of a month, you will need create blank log files for the previous days. e.g.:
  - `cd /usr/local/glassfish4/glassfish/domains/domain1/logs`
  - `touch counter_2019-02-01.log`
  - ...
  - `touch counter_2019-02-20.log`
- Run Counter Processor.
  - `CONFIG_FILE=counter-processor-config.yaml python36 main.py`
  - A JSON file in SUSHI format will be created in the directory you specified under “output\_file” in the config file.
- Populate views and downloads for your datasets based on the SUSHI JSON file. The “/tmp” directory is used in the example below.
  - `curl -X POST "http://localhost:8080/api/admin/makeDataCount/addUsageMetricsFromSushiReport?reportOnDisk=/tmp/make-data-count-report.json"`
- Verify that views and downloads are available via API.
  - Now that views and downloads have been recorded in the Dataverse database, you should make sure you can retrieve them from a dataset or two. Use the *Dataset Metrics* endpoints in the *Native API* section of the API Guide.

## Populate Views and Downloads Nightly

Running `main.py` to create the SUSHI JSON file and the subsequent calling of the Dataverse API to process it should be added as a cron job.

## Sending Usage Metrics to the DataCite Hub

Once you are satisfied with your testing, you should contact [support@datacite.org](mailto:support@datacite.org) for your JSON Web Token and change “upload\_to\_hub” to “True” in the config file. The next time you run `main.py` the following metrics will be sent to the DataCite hub for each published dataset:

- Views (“investigations” in COUNTER)
- Downloads (“requests” in COUNTER)

## 2.8.5 Configuring Dataverse for Make Data Count Citations

Please note: as explained in the note above about limitations, this feature is not available to installations of Dataverse that use Handles.

Please note that in the curl example, Bash environment variables are used with the idea that you can set a few environment variables and copy and paste the examples as is. For example, “\$DOI” could become “doi:10.5072/FK2/BL2IBM” by issuing the following export command from Bash:

```
export DOI="doi:10.5072/FK2/BL2IBM"
```

To confirm that the environment variable was set properly, you can use echo like this:

```
echo $DOI
```

On some periodic basis (perhaps weekly) you should call the following curl command for each published dataset to update the list of citations that have been made for that dataset.

```
curl -X POST "http://localhost:8080/api/admin/makeDataCount/:persistentId/updateCitationsForDataset?persistentId=$DOI"
```

Citations will be retrieved for each published dataset and recorded in the Dataverse database.

For how to get the citations out of Dataverse, see “Retrieving Citations for a Dataset” under *Dataset Metrics* in the *Native API* section of the API Guide.

Please note that while Dataverse has a metadata field for “Related Dataset” this information is not currently sent as a citation to Crossref.

### 2.8.6 Retrieving Make Data Count Metrics from the DataCite Hub

The following metrics can be downloaded directly from the DataCite hub (see <https://support.datacite.org/docs/eventdata-guide>) for datasets hosted by Dataverse installations that have been configured to send these metrics to the hub:

- Total Views for a Dataset
- Unique Views for a Dataset
- Total Downloads for a Dataset
- Downloads for a Dataset
- Citations for a Dataset (via Crossref)

### 2.8.7 Retrieving Make Data Count Metrics from Dataverse

The Dataverse API endpoints for retrieving Make Data Count metrics are described below under *Dataset Metrics* in the *Native API* section of the API Guide.

Please note that it is also possible to retrieve metrics from the DataCite hub itself via <https://api.datacite.org>

## 2.9 Integrations

Now that you’ve installed Dataverse, you might want to set up some integrations with other systems. Many of these integrations are open source and are cross listed in the *Apps* section of the API Guide.

### Contents:

- *Getting Data In*
  - *Dropbox*

- *Open Science Framework (OSF)*
- *RSpace*
- *Open Journal Systems (OJS)*
- *Analysis and Computation*
  - *Data Explorer*
  - *TwoRavens/Zelig*
  - *WorldMap*
  - *Compute Button*
  - *Whole Tale*
- *Discoverability*
  - *OAI-PMH (Harvesting)*
  - *SHARE*
- *Research Data Preservation*
  - *Archivematica*
  - *DuraCloud/Chronopolis*
- *Future Integrations*

### 2.9.1 Getting Data In

A variety of integrations are oriented toward making it easier for your researchers to deposit data into your installation of Dataverse.

#### Dropbox

If your researchers have data on Dropbox, you can make it easier for them to get it into Dataverse by setting the `dataverse.dropbox.key` JVM option described in the *Configuration* section of the Installation Guide.

#### Open Science Framework (OSF)

The Center for Open Science’s Open Science Framework (OSF) is an open source software project that facilitates open collaboration in science research across the lifespan of a scientific project.

For instructions on depositing data from OSF to your installation of Dataverse, your researchers can visit <http://help.osf.io/m/addons/l/863978-connect-dataverse-to-a-project>

#### RSpace

RSpace is an affordable and secure enterprise grade electronic lab notebook (ELN) for researchers to capture and organize data.

For instructions on depositing data from RSpace to your installation of Dataverse, your researchers can visit <https://www.researchspace.com/help-and-support-resources/dataverse-integration/>

## Open Journal Systems (OJS)

Open Journal Systems (OJS) is a journal management and publishing system that has been developed by the Public Knowledge Project to expand and improve access to research.

The OJS Dataverse Plugin adds data sharing and preservation to the OJS publication process.

As of this writing only OJS 2.x is supported and instructions for getting started can be found at [https://github.com/pkp/ojs/tree/ojs-stable-2\\_4\\_8/plugins/generic/dataverse](https://github.com/pkp/ojs/tree/ojs-stable-2_4_8/plugins/generic/dataverse)

If you are interested in OJS 3.x supporting deposit from Dataverse, please leave a comment on <https://github.com/pkp/pkp-lib/issues/1822>

## 2.9.2 Analysis and Computation

### Data Explorer

Data Explorer is a GUI which lists the variables in a tabular data file allowing searching, charting and cross tabulation analysis.

For installation instructions, see the *External Tools* section.

### TwoRavens/Zelig

TwoRavens is a web application for tabular data exploration and statistical analysis with Zelig.

For installation instructions, see the *External Tools* section.

### WorldMap

WorldMap helps researchers visualize and explore geospatial data by creating maps.

For installation instructions, see *Geoconnect and WorldMap*.

### Compute Button

The “Compute” button is still highly experimental and has special requirements such as use of a Swift object store, but it is documented under “Setting up Compute” in the *Configuration* section of the Installation Guide.

### Whole Tale

Whole Tale enables researchers to analyze data using popular tools including Jupyter and RStudio with the ultimate goal of supporting publishing of reproducible research packages. Users can [import data from Dataverse](#) via identifier (e.g., DOI, URI, etc) or through the External Tools integration. For installation instructions, see the *External Tools* section or the *Integration* section of the Whole Tale User Guide.

## 2.9.3 Discoverability

Integration with [DataCite](#) is built in to Dataverse. When datasets are published, metadata is sent to DataCite. You can further increase the discoverability of your datasets by setting up additional integrations.



## OAI-PMH (Harvesting)

Dataverse supports a protocol called OAI-PMH that facilitates harvesting datasets from one system into another. For details on harvesting, see the *Managing Harvesting Server and Sets* section.

## SHARE

SHARE is building a free, open, data set about research and scholarly activities across their life cycle. It's possible to add and installation of Dataverse as one of the *sources* they include if you contact the SHARE team.

## 2.9.4 Research Data Preservation

### Archivematica

Archivematica is an integrated suite of open-source tools for processing digital objects for long-term preservation, developed and maintained by Artefactual Systems Inc. Its configurable workflow is designed to produce system-independent, standards-based Archival Information Packages (AIPs) suitable for long-term storage and management.

Sponsored by the [Ontario Council of University Libraries \(OCUL\)](#), this technical integration enables users of Archivematica to select datasets from connected Dataverse instances and process them for long-term access and digital preservation. For more information and list of known issues, please refer to Artefactual's [release notes](#), [integration documentation](#), and the [project wiki](#).

### DuraCloud/Chronopolis

Dataverse can be configured to submit a copy of published Datasets, packaged as [Research Data Alliance conformant](#) zipped [BagIt](#) bags to the [Chronopolis](#) via [DuraCloud](#)

For details on how to configure this integration, look for "DuraCloud/Chronopolis" in the *Configuration* section of the *Installation Guide*.

## 2.9.5 Future Integrations

The [Dataverse roadmap](#) is a good place to see integrations that the core Dataverse team is working on.

The [Community Dev](#) column of our project board is a good way to track integrations that are being worked on by the Dataverse community but many are not listed and if you have an idea for an integration, please ask on the [dataverse-community](#) mailing list if someone is already working on it.

Many integrations take the form of "external tools". See the *External Tools* section for details. External tool makers should check out the *Building External Tools* section of the *API Guide*.

Please help us keep this page up to date making a pull request! To get started, see the *Writing Documentation* section of the *Developer Guide*.

## 2.10 Geoconnect and WorldMap

One of the optional components listed under "Architecture and Components" in the *Preparation* section of the *Installation Guide* is [Geoconnect](#), a piece of middleware that allows Dataverse users to create maps in [WorldMap](#) based on geospatial data stored in Dataverse. For more details on the feature from the user perspective, see the *WorldMap: Geospatial Data Exploration* section of the *User Guide*.

**Contents:**

- *Update “mapitlink”*
- *Removing Dead Explore Links*

## 2.10.1 Update “mapitlink”

SQL commands to point a Dataverse installation at different Geoconnect servers:

**Geoconnect Production** *geoconnect.datascience.iq.harvard.edu*

```
update worldmapauth_tokentype set mapitlink = 'https://geoconnect.datascience.iq.harvard.edu/shapefile/map-it', hostname='geoconnect.datascience.iq.harvard.edu'
↳ where name = 'GEOCONNECT';
```

**Heroku Test** *geoconnect-dev.herokuapp.com*

```
update worldmapauth_tokentype set mapitlink = 'https://geoconnect-dev.herokuapp.com/shapefile/map-it', hostname='geoconnect-dev.herokuapp.com'
↳ where name = 'GEOCONNECT';
```

### View Current Settings

```
SELECT * from worldmapauth_tokentype;
```

## 2.10.2 Removing Dead Explore Links

After a map has been created in WorldMap (assuming all the setup has been done), an “Explore” button will appear next to the name of the file in Dataverse. The “Explore” button should open the map in WorldMap. In rare occasions, the map has been deleted on the WorldMap side such that the “Explore” button goes nowhere, resulting in a dead link, a 404.

Functionality has been added on the Dataverse side to iterate through all the maps Dataverse knows about (stored in the `maplayermetadata` database table) and to check for the existence of each map in WorldMap. The status code returned from WorldMap (200, 404, etc.) is recorded in Dataverse along with a timestamp of when the check was performed. To perform this check, you can execute the following `curl` command:

```
curl -X POST http://localhost:8080/api/admin/geoconnect/mapLayerMetadatas/check
```

The output above will contain the `layerLink` being checked as well as the HTTP response status code (200, 404, etc.) in the `lastVerifiedStatus` field. 200 means OK and 404 means not found. 500 might indicate that the map is only temporarily unavailable. The `lastVerifiedStatus` and `lastVerifiedTime` will be persisted to the `maplayermetadata` database table.

Armed with this information about WorldMap returning a 404 for a map, you may want to delete any record of the map on the Dataverse side so that the “Explore” button goes away (and so that thumbnail files are cleaned up). To accomplish this, use the following `curl` command, substituting the id of the file:

```
curl -H "X-Dataverse-key: $API_TOKEN" -X DELETE http://localhost:8080/api/files/{file_id}/map
```

End users can also run the `DELETE` command above (if they have permission to edit the dataset) but it’s more likely that the sysadmin reading this guide will run it for them. It is recommended that you add the “check” command above

to cron so that Dataverse periodically checks on the status of maps in WorldMap. In addition to the “check all maps” command above, you can also check an individual map with the following `curl` command, substituting the id of the row from the `maplayermetadata` table:

```
curl -X POST http://localhost:8080/api/admin/geoconnect/mapLayerMetadatas/  
check/{id}
```

## 2.11 User Administration

This section focuses on user administration tools and tasks.

### Contents:

- *Manage Users*
  - *List Users via API*
- *Merge User Accounts*
- *Change User Identifier*
- *Confirm Email*
- *Deleting an API Token*

### 2.11.1 Manage Users

The Manage Users table gives the network administrator a list of all user accounts in table form. You can access it by clicking the “Manage Users” button on the *Dashboard*, which is linked from the header of all Dataverse pages (if you’re logged in as an administrator). It lists username, full name, email address, affiliation, the authentication method they use, the roles their account has been granted, and whether or not they have Superuser status.

Users are listed alphabetically by username. The search bar above the table allows you to search for a specific user. It performs a right-truncated wildcard search of the Username, Name, and Email columns. This means, if you search “baseba” then it will search those three columns for any string of text that begins with “baseba”, e.g. “baseball” or “baseballfan”.

If you would like to assign or remove a user’s Superuser status, then you can do so by checking or unchecking their checkbox under the “Superuser” column.

If you would like to remove all roles/permissions from a user’s account (in the event of their leaving your organization, for example) then you can do so by clicking the “Remove All” button under the Roles column. This will keep the user’s account active, but will revert it to put the account on the level of a default user with default permissions.

### List Users via API

There are two ways to list users via API. If you have relatively few users, you can get them all as a dump with this command with a superuser API token:

```
curl -H "X-Dataverse-key: $API_TOKEN" http://localhost:8080/api/admin/  
↪authenticatedUsers
```

If you have many users and want to be able to search and paginate through the results, use the command below with a superuser API token:

```
curl -H "X-Dataverse-key: $API_TOKEN" http://localhost:8080/api/admin/list-users
```

With the `list-users` form you can include the following optional query parameters:

- `searchTerm` A string that matches the beginning of a user identifier, first name, last name or email address.
- `itemsPerPage` The number of detailed results to return. The default is 25. This number has no limit. e.g. You could set it to 1000 to return 1,000 results
- `selectedPage` The page of results to return. The default is 1.

## 2.11.2 Merge User Accounts

See *Merge User Accounts*

## 2.11.3 Change User Identifier

See *Change User Identifier*

## 2.11.4 Confirm Email

Dataverse encourages builtin/local users to verify their email address upon signup or email change so that sysadmins can be assured that users can be contacted.

The app will send a standard welcome email with a URL the user can click, which, when activated, will store a `lastconfirmed` timestamp in the `authenticateduser` table of the database. Any time this is “null” for a user (immediately after signup and/or changing of their Dataverse email address), their current email on file is considered to not be verified. The link that is sent expires after a time (the default is 24 hours), but this is configurable by a superuser via the `:MinutesUntilConfirmEmailTokenExpires` config option.

Should users’ URL token expire, they will see a “Verify Email” button on the account information page to send another URL.

Sysadmins can determine which users have verified their email addresses by looking for the presence of the value `emailLastConfirmed` in the JSON output from listing users (see the “Admin” section of the *Native API*). As mentioned in the *Account Creation + Management* section of the User Guide, the email addresses for Shibboleth users are re-confirmed on every login.

## 2.11.5 Deleting an API Token

If an API token is compromised it should be deleted. Users can generate a new one for themselves as explained in the *Account Creation + Management* section of the User Guide, but you may want to preemptively delete tokens from the database.

Using the API token `7ae33670-be21-491d-a244-008149856437` as an example:

```
delete from apitoken where tokenstring = '7ae33670-be21-491d-a244-008149856437';
```

You should expect the output `DELETE 1` after issuing the command above.

## 2.12 Managing Datasets and Dataverses

### Contents:

- *Dataverses*
  - *Delete a Dataverse*
  - *Move a Dataverse*
  - *Link a Dataverse*
  - *Unlink a Dataverse*
  - *Add Dataverse RoleAssignments to Child Dataverses*
- *Datasets*
  - *Move a Dataset*
  - *Link a Dataset*
  - *Unlink a Dataset*
  - *Mint a PID for a File That Does Not Have One*
  - *Mint PIDs for Files That Do Not Have Them*
  - *Mint a New DOI for a Dataset with a Handle*
  - *Send Dataset metadata to PID provider*
  - *Make Metadata Updates Without Changing Dataset Version*
  - *Diagnose Constraint Violations Issues in Datasets*

### 2.12.1 Dataverses

#### Delete a Dataverse

Dataverses have to be empty to delete them. Navigate to the dataverse and click “Edit” and then “Delete Dataverse” to delete it. To delete a dataverse via API, see the *Native API* section of the API Guide.

#### Move a Dataverse

Moves a dataverse whose id is passed to a new dataverse whose id is passed. The dataverse alias also may be used instead of the id. If the moved dataverse has a guestbook, template, metadata block, link, or featured dataverse that is not compatible with the destination dataverse, you will be informed and given the option to force the move and remove the association. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST http://$SERVER/api/dataverses/$id/move/
↪$destination-id
```

#### Link a Dataverse

Creates a link between a dataverse and another dataverse (see the Linked Dataverses + Linked Datasets section of the *Dataverse Management* guide for more information). Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT http://$SERVER/api/dataverses/$linked-  
↳dataverse-alias/link/$linking-dataverse-alias
```

### Unlink a Dataverse

Removes a link between a dataverse and another dataverse. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X DELETE http://$SERVER/api/dataverses/$linked-  
↳dataverse-alias/deleteLink/$linking-dataverse-alias
```

### Add Dataverse RoleAssignments to Child Dataverses

Recursively assigns the users and groups having a role(s), that are in the set configured to be inheritable via the `:Inherit-ParentRoleAssignments` setting, on a specified dataverse to have the same role assignments on all of the dataverses that have been created within it. The response indicates success or failure and lists the individuals/groups and dataverses involved in the update. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" http://$SERVER/api/admin/dataverse/$dataverse-  
↳alias//addRoleAssignmentsToChildren
```

## 2.12.2 Datasets

### Move a Dataset

Superusers can move datasets using the dashboard. See also [Dashboard](#).

Moves a dataset whose id is passed to a dataverse whose alias is passed. If the moved dataset has a guestbook or a dataverse link that is not compatible with the destination dataverse, you will be informed and given the option to force the move (with `forceMove=true` as a query parameter) and remove the guestbook or link (or both). Only accessible to users with permission to publish the dataset in the original and destination dataverse.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST http://$SERVER/api/datasets/$id/move/  
↳$alias
```

### Link a Dataset

Creates a link between a dataset and a dataverse (see the [Linked Dataverses + Linked Datasets](#) section of the [Dataverse Management](#) guide for more information).

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT http://$SERVER/api/datasets/$linked-  
↳dataset-id/link/$linking-dataverse-alias
```

### Unlink a Dataset

Removes a link between a dataset and a dataverse. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X DELETE http://$SERVER/api/datasets/$linked-  
↳dataset-id/deleteLink/$linking-dataverse-alias
```

## Mint a PID for a File That Does Not Have One

In the following example, the database id of the file is 42:

```
export FILE_ID=42
curl http://localhost:8080/api/admin/$FILE_ID/registerDataFile
```

## Mint PIDs for Files That Do Not Have Them

If you have a large number of files, you might want to consider minting PIDs for files individually using the `registerDataFile` endpoint above in a for loop, sleeping between each registration:

```
curl http://localhost:8080/api/admin/registerDataFileAll
```

## Mint a New DOI for a Dataset with a Handle

Mints a new identifier for a dataset previously registered with a handle. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST http://$SERVER/api/admin/$dataset-id/
↳ reregisterHDLToPID
```

## Send Dataset metadata to PID provider

Forces update to metadata provided to the PID provider of a published dataset. Only accessible to superusers.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST http://$SERVER/api/datasets/$dataset-id/
↳ modifyRegistrationMetadata
```

## Make Metadata Updates Without Changing Dataset Version

As a superuser, click “Update Current Version” when publishing. (This option is only available when a ‘Minor’ update would be allowed.)

## Diagnose Constraint Violations Issues in Datasets

To identify invalid data values in specific datasets (if, for example, an attempt to edit a dataset results in a `ConstraintViolationException` in the server log), or to check all the datasets in the Dataverse for constraint violations, see *Dataset Validation* in the *Native API* section of the User Guide.

## 2.13 Solr Search Index

Dataverse requires Solr to be operational at all times. If you stop Solr, you should see a error about this on the root dataverse page, which is powered by the search index Solr provides. You can set up Solr by following the steps in our Installation Guide’s *Prerequisites* and *Configuration* sections explaining how to configure it. This section you’re reading now is about the care and feeding of the search index. PostgreSQL is the “source of truth” and the Dataverse application will copy data from PostgreSQL into Solr. For this reason, the search index can be rebuilt at any time. Depending on the amount of data you have, this can be a slow process. You are encouraged to experiment with production data to get a sense of how long a full reindexing will take.

**Contents:**

- *Full Reindex*
  - *Clear and Reindex*
    - \* *Clearing Data from Solr*
    - \* *Start Async Reindex*
  - *Reindex in Place*
    - \* *Clear Index Timestamps*
    - \* *Start or Continue Async Reindex*
- *Manual Reindexing*
  - *Reindexing Dataverses*
  - *Reindexing Datasets*
- *Manually Querying Solr*

### 2.13.1 Full Reindex

There are two ways to perform a full reindex of the Dataverse search index. Starting with a “clear” ensures a completely clean index but involves downtime. Reindexing in place doesn’t involve downtime but does not ensure a completely clean index.

#### Clear and Reindex

##### Clearing Data from Solr

Please note that the moment you issue this command, it will appear to end users looking at the root dataverse page that all data is gone! This is because the root dataverse page is powered by the search index.

```
curl http://localhost:8080/api/admin/index/clear
```

##### Start Async Reindex

Please note that this operation may take hours depending on the amount of data in your system. This known issue is being tracked at <https://github.com/IQSS/dataverse/issues/50>

```
curl http://localhost:8080/api/admin/index
```

##### Reindex in Place

An alternative to completely clearing the search index is to reindex in place.

##### Clear Index Timestamps

```
curl -X DELETE http://localhost:8080/api/admin/index/timestamps
```



## Start or Continue Async Reindex

If indexing stops, this command should pick up where it left off based on which index timestamps have been set, which is why we start by clearing these timestamps above. These timestamps are stored in the `dvobject` database table.

```
curl http://localhost:8080/api/admin/index/continue
```

## 2.13.2 Manual Reindexing

If you have made manual changes to a dataset in the database or wish to reindex a dataset that solr didn't want to index properly, it is possible to manually reindex dataverses and datasets.

### Reindexing Dataverses

Dataverses must be referenced by database object ID. If you have direct database access an SQL query such as

```
select id from dataverse where alias='datavarsealias';
```

should work, or you may click the Dataverse's "Edit" menu and look for `dataverseId=` in the URLs produced by the drop-down. Then, to re-index:

```
curl http://localhost:8080/api/admin/index/dataverses/135
```

which should return: `_{“status”:“OK”,“data”:{“message”:“starting reindex of dataverse 135”}}_`

### Reindexing Datasets

Datasets may be referenced by persistent ID or by database object ID. To re-index by persistent ID:

```
curl http://localhost:8080/api/admin/index/dataset?persistentId=doi:10.5072/FK2/AAA000
```

To re-index a dataset by its database ID:

```
curl http://localhost:8080/api/admin/index/datasets/7504557
```

## 2.13.3 Manually Querying Solr

If you suspect something isn't indexed properly in solr, you may bypass the Dataverse web interface and query the command line directly to verify what solr returns:

```
curl "http://localhost:8983/solr/collection1/select?q=dsPersistentId:doi:10.15139/S3/HFV0AO"
```

to see the JSON you were hopefully expecting to see passed along to Dataverse.

## 2.14 IP Groups

IP Groups can be used to permit download of restricted files by IP addresses rather than people. For example, you may want to allow restricted files to be downloaded by researchers who physically enter a library and make use of the library's network.

**Contents:**

- *Listing IP Groups*
- *Creating an IP Group*
- *Listing an IP Group*
- *Deleting an IP Group*

### 2.14.1 Listing IP Groups

IP Groups can be listed with the following curl command:

```
curl http://localhost:8080/api/admin/groups/ip
```

### 2.14.2 Creating an IP Group

IP Groups must be expressed as ranges in IPv4 or IPv6 format. For illustrative purposes, here is an example of the entire IPv4 and IPv6 range that you can download and edit to have a narrower range to meet your needs. If you need your IP Group to only encompass a single IP address, you must enter that IP address for the “start” and “end” of the range. If you don’t use IPv6 addresses, you can delete that section of the JSON. Please note that the “alias” must be unique if you define multiple IP Groups. You should give it a meaningful “name” since both “alias” and “name” will appear and be searchable in the GUI when your users are assigning roles.

```
{
  "alias": "ipGroupAll",
  "name": "IP group to match all IPv4 and IPv6 addresses",
  "ranges": [
    [
      "0.0.0.0",
      "255.255.255.255"
    ],
    [
      "::",
      "ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff"
    ]
  ]
}
```

Let’s say you download the example above and edit it to give it a range used by your library, giving it a filename of ipGroup1.json and putting it in the /tmp directory. Next, load it into Dataverse using the following curl command:

```
curl -X POST -H 'Content-type: application/json' http://localhost:8080/api/admin/groups/ip --upload-file /tmp/ipGroup1.json
```

Note that you can update a group the same way, as long as you use the same alias.

### 2.14.3 Listing an IP Group

Let’s say you used “ipGroup1” as the alias of the IP Group you created above. To list just that IP Group, you can include the alias in the curl command like this:

```
curl http://localhost:8080/api/admin/groups/ip/ipGroup1
```

### 2.14.4 Deleting an IP Group

It is not recommended to delete an IP Group that has been assigned roles. If you want to delete an IP Group, you should first remove its permissions.

To delete an IP Group with an alias of “ipGroup1”, use the curl command below:

```
curl -X DELETE http://localhost:8080/api/admin/groups/ip/ipGroup1
```

## 2.15 Monitoring

Once you’re in production, you’ll want to set up some monitoring. This page may serve as a starting point for you but you are encouraged to share your ideas with the Dataverse community!

### Contents:

- *Operating System Monitoring*
  - *Munin*
- *HTTP Traffic*
  - *Monitoring HTTP Traffic from the Client Side*
  - *Monitoring HTTP Traffic from the Server Side*
  - *AWStats*
- *Database Connection Pool used by Glassfish*
- *actionlogrecord*
- *Edit Draft Versions Logging*
- *Solr Indexing Failures Logging*
- *EJB Timers*

### 2.15.1 Operating System Monitoring

In production you’ll want to monitor the usual suspects such as CPU, memory, free disk space, etc. There are a variety of tools in this space but we’ll highlight Munin below because it’s relatively easy to set up.

#### Munin

<http://munin-monitoring.org> says, “A default installation provides a lot of graphs with almost no work.” From RHEL or CentOS 7, you can try the following steps.

Enable the EPEL yum repo (if you haven’t already):

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

Install Munin:

```
yum install munin
```

Start Munin:

```
systemctl start munin-node.service
```

Configure Munin to start at boot:

```
systemctl enable munin-node.service
```

Create a username/password (i.e. “admin” for both):

```
htpasswd /etc/munin/munin-htpasswd admin
```

Assuming you are fronting Glassfish with Apache, prevent Apache from proxying “/munin” traffic to Glassfish by adding the following line to your Apache config:

```
ProxyPassMatch ^/munin !
```

Then reload Apache to pick up the config change:

```
systemctl reload httpd.service
```

Test auth for the web interface:

```
curl http://localhost/munin/ -u admin:admin
```

At this point, graphs should start being generated for disk, network, processes, system, etc.

### 2.15.2 HTTP Traffic

HTTP traffic can be monitored from the client side, the server side, or both.

#### Monitoring HTTP Traffic from the Client Side

HTTP traffic for web clients that have cookies enabled (most browsers) can be tracked by Google Analytics (<https://www.google.com/analytics/>) and Matomo (formerly “Piwik”; <https://matomo.org/>) as explained in the *Web Analytics Code* section of the Installation Guide.

To track analytics beyond pageviews, style classes have been added for end user action buttons, which include:

```
btn-compute, btn-contact, btn-download, btn-explore, btn-export, btn-preview,  
btn-request, btn-share
```

#### Monitoring HTTP Traffic from the Server Side

There are a wide variety of solutions available for monitoring HTTP traffic from the server side. The following are merely suggestions and a pull request against what is written here to add additional ideas is certainly welcome! Are you excited about the ELK stack (Elasticsearch, Logstash, and Kibana)? The TICK stack (Telegraph InfluxDB Chronograph and Kapacitor)? GoAccess? Prometheus? Graphite? Splunk? Please consider sharing your work with the Dataverse community!

#### AWStats

AWStats is a venerable tool for monitoring web traffic based on Apache access logs. On RHEL/CentOS 7, you can try the following steps.

Enable the EPEL yum repo:

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.  
noarch.rpm
```

Install AWStats:

```
yum install awstats
```

Assuming you are using HTTPS rather than HTTP (and you should!), edit `/etc/awstats/awstats.standalone.conf` and change `LogFile="/var/log/httpd/access_log"` to `LogFile="/var/log/httpd/ssl_access_log"`. In the same file, change `LogFormat=1` to `LogFormat=4`. Make both of these changes (`LogFile` and `LogFormat` in `/etc/awstats/awstats.localhost.localdomain.conf` as well).

Process the logs:

```
/usr/share/awstats/tools/awstats_updateall.pl now
```

Please note that load balancers (such as Amazon's ELB) might interfere with the `LogFormat` mentioned above. To start troubleshooting errors such as AWStats did not find any valid log lines that match your `LogFormat` parameter, you might need to bump up the value of `NbOfLinesForCorruptedLog` in the config files above and re-try while you iterate on your Apache and AWStats config.

Please note that the Dataverse team has attempted to parse Glassfish logs using AWStats but it didn't seem to just work and posts have been made at <https://stackoverflow.com/questions/49134154/what-logformat-definition-does-awstats-require-to-parse-glassfish-http-access-logs> and <https://sourceforge.net/p/awstats/discussion/43428/thread/9b1befda/> that can be followed up on some day.

### 2.15.3 Database Connection Pool used by Glassfish

<https://github.com/IQSS/dataverse/issues/2595> contains some information on enabling monitoring of Glassfish, which is disabled by default. It's a TODO to document what to do here if there is sufficient interest.

### 2.15.4 actionlogrecord

There is a database table called `actionlogrecord` that captures events that may be of interest. See <https://github.com/IQSS/dataverse/issues/2729> for more discussion around this table.

### 2.15.5 Edit Draft Versions Logging

Changes made to draft versions of datasets are logged in a folder called `logs/edit-drafts`. See <https://github.com/IQSS/dataverse/issues/5145> for more information on this logging.

### 2.15.6 Solr Indexing Failures Logging

Failures occurring during the indexing of dataverses and datasets are logged in a folder called `logs/process-failures`. This logging will include instructions for manually re-running the failed processes. It may be advantageous to set up a automatic job to monitor new entries into this log folder so that indexes could be re-run.

### 2.15.7 EJB Timers

Should you be interested in monitoring the EJB timers, this script may be used as an example:

```
#!/usr/bin/env bash

# example monitoring script for EBJ timers.
# currently assumes that there are two timers
# real monitoring commands should replace the echo statements for production use

r0=`curl -s http://localhost:8080/ejb-timer-service-app/timer`

if [ $? -ne 0 ]; then
    echo "alert - no timer service" # put real alert command here
fi

r1=`echo $r0 | grep -c "There are 2 active persistent timers on this container"`

if [ "1" -ne "$r1" ]; then
    echo "alert - no active timers" # put real alert command here
fi
```

## 2.16 Reporting Tools

Reporting tools created by members of the Dataverse community.

- Matrix (<<https://github.com/rindataverse/matrix>>): Collaboration Matrix is a visualization showing the connectedness and collaboration between authors and their affiliations. Visit <https://rin.lipi.go.id/matrix/> to play with a production installation.
- Dataverse Web Report (<<https://github.com/scholarsportal/Dataverse-Web-Report>>): Creates interactive charts showing data extracted from the Dataverse Excel Report
- Dataverse Reports for Texas Digital Library (<<https://github.com/TexasDigitalLibrary/dataverse-reports>>): A python3-based tool to generate and email statistical reports from Dataverse (<https://dataverse.org/>) using the native API and database queries.
- dataverse-metrics (<<https://github.com/IQSS/dataverse-metrics>>): Aggregates and visualizes metrics for installations of Dataverse around the world or a single Dataverse installation.

## 2.17 Maintenance

When you have scheduled down time for your production servers, we provide a `sample maintenance page` for you to use. To download, right-click and select “Save Link As”.

The maintenance page is intended to be a static page served by Apache to provide users with a nicer, more informative experience when the site is unavailable.

## 2.18 Backups

Running tape, or similar backups to ensure the long term preservation of all the data stored in the Dataverse is an implied responsibility that should be taken most seriously.

*In addition* to running these disk-level backups, we have provided an experimental script that can be run on schedule (via a cron job or something similar) to create extra archival copies of all the Datafiles stored in the Dataverse

on a remote storage server, accessible via an ssh connection. The script and some documentation can be found in `scripts/backup/run_backup` in the Dataverse source tree at <https://github.com/IQSS/dataverse>. Some degree of knowledge of system administration and Python is required.

Once again, the script is experimental and NOT a replacement of regular and reliable disk backups!

## 2.19 Troubleshooting

Sometimes Dataverse users get into trouble. Sometimes Dataverse itself gets into trouble. If something has gone wrong, this section is for you.

### Contents:

- *Using Dataverse APIs to Troubleshoot and Fix Problems*
  - *A Dataset Is Locked And Cannot Be Edited or Published*
  - *Someone Created Spam Datasets and I Need to Delete Them*
  - *A User Needs Their Account to Be Converted From Institutional (Shibboleth), ORCID, Google, or GitHub to Something Else*
- *Glassfish*
- *Deployment fails, “EJB Timer Service not available”*
- *Timer not working*
- *Constraint Violations Issues*
- *Many Files with a File Type of “Unknown”, “Application”, or “Binary”*

### 2.19.1 Using Dataverse APIs to Troubleshoot and Fix Problems

See the *Introduction* section of the API Guide for a high level overview of Dataverse APIs. Below are listed problems that support teams might encounter that can be handled via API (sometimes only via API).

#### A Dataset Is Locked And Cannot Be Edited or Published

It's normal for the ingest process described in the *Tabular Data, Representation, Storage and Ingest* section of the User Guide to take some time but if hours or days have passed and the dataset is still locked, you might want to inspect the locks and consider deleting some or all of them.

See *Managing Datasets and Dataverses*.

#### Someone Created Spam Datasets and I Need to Delete Them

Depending on how open your installation of Dataverse is to the general public creating datasets, you may sometimes need to deal with spam datasets.

Look for “destroy” in the *Native API* section of the API Guide.

## A User Needs Their Account to Be Converted From Institutional (Shibboleth), ORCID, Google, or GitHub to Something Else

See *Converting Shibboleth Users to Local* and *Converting OAuth Users to Local*.

### 2.19.2 Glassfish

`server.log` is the main place to look when you encounter problems. Hopefully an error message has been logged. If there's a stack trace, it may be of interest to developers, especially they can trace line numbers back to a tagged version.

For debugging purposes, you may find it helpful to increase logging levels as mentioned in the *Debugging* section of the Developer Guide.

Our guides focus on using the command line to manage Glassfish but you might be interested in an admin GUI at <http://localhost:4848>

### 2.19.3 Deployment fails, “EJB Timer Service not available”

Sometimes the Dataverse application fails to deploy, or Glassfish fails to restart once the application is deployed, with the following error message: “remote failure: Error occurred during deployment: Exception while loading the app : EJB Timer Service is not available. Please see server.log for more details.”

We don't know what's causing this issue, but here's a known workaround:

- Stop Glassfish;
- Remove the `generated` and `osgi-cache` directories;
- Delete all the rows from the `EJB__TIMER__TBL` table in the database;
- Start Glassfish

The shell script below performs the steps above. Note that it may or may not work on your system, so it is provided as an example only, downloadable [here](#). Aside from the configuration values that need to be changed to reflect your environment (the Glassfish directory, name of the database, etc.) the script relies on the database being configured in a certain way for access. (See the comments in the script for more information)

```
#!/bin/sh

# EBJ timers sometimes cause problems; utility to clear generated directories and_
↪database rows

# assumes this script is being run as root, and that the postgres user had_
↪passwordless
# access to the database (local sockets, or appropriate environmental variables).

# will restart glassfish if it's stopped; comment out the `start-domain` command at_
↪the end
# if you'd like to avoid that.

# directory where glassfish is installed
GLASSFISH_DIR=/usr/local/glassfish4

# directory within glassfish (defaults)
DV_DIR=${GLASSFISH_DIR}/glassfish/domains/domain1

# name of dataverse database
```



```
DV_DB=dvndb

# OS user for the database
DB_USER=postgres

# stop the glassfish domain (generates a warning if glassfish is stopped)
${GLASSFISH_DIR}/bin/asadmin stop-domain

rm -rf ${GLASSFISH_DIR}/${DV_DIR}/generated/
rm -rf ${GLASSFISH_DIR}/${DV_DIR}/osgi-cache/felix

sudo -u ${DB_USER} psql ${DV_DB} -c 'delete from "EJB__TIMER__TBL"';

# restart the domain (also generates a warning if glassfish is stopped)
${GLASSFISH_DIR}/bin/asadmin start-domain
```

### 2.19.4 Timer not working

Dataverse relies on EJB timers to perform scheduled tasks: harvesting from remote servers, updating the local OAI sets and running metadata exports. (See *Dataverse Application Timers* for details.) If these scheduled jobs are not running on your server, this may be the result of the incompatibility between the version of PostgreSQL database you are using, and PostgreSQL JDBC driver in use by your instance of Glassfish. The symptoms:

If you are seeing the following in your server.log...

Handling timeout on ...

followed by an Exception stack trace with these lines in it:

Internal Exception: java.io.StreamCorruptedException: invalid stream header ...

Exception Description: Could not deserialize object from byte array ...

... it most likely means that it is the JDBC driver incompatibility that's preventing the timer from working correctly. Make sure you install the correct version of the driver. For example, if you are running the version 9.3 of PostgreSQL, make sure you have the driver postgresql-9.3-1104.jdbc4.jar in your <GLASSFISH FOLDER>/glassfish/lib directory. Go [here](#) to download the correct version of the driver. If you have an older driver in glassfish/lib, make sure to remove it, replace it with the new version and restart Glassfish. (You may need to remove the entire contents of <GLASSFISH FOLDER>/glassfish/domains/domain1/generated before you start Glassfish).

### 2.19.5 Constraint Violations Issues

In real life production use, it may be possible to end up in a situation where some values associated with the datasets in your database are no longer valid under the constraints enforced by the latest version of Dataverse. This is not very likely to happen, but if it does, the symptoms will be as follows: Some datasets can no longer be edited, long exception stack traces logged in the Glassfish server log, caused by:

```
javax.validation.ConstraintViolationException:
Bean Validation constraint(s) violated while executing Automatic Bean Validation on
↳callback event:'preUpdate'.
Please refer to embedded ConstraintViolations for details.
```

(contrary to what the message suggests, there are no specific “details” anywhere in the stack trace that would explain what values violate which constraints)

To identify the specific invalid values in the affected datasets, or to check all the datasets in the Dataverse for constraint violations, see [Dataset Validation](#) in the [Native API](#) section of the User Guide.

### **2.19.6 Many Files with a File Type of “Unknown”, “Application”, or “Binary”**

From the home page of a Dataverse installation you can get a count of files by file type by clicking “Files” and then scrolling down to “File Type”. If you see a lot of files that are “Unknown”, “Application”, or “Binary” you can have Dataverse attempt to redetect the file type by using the [Redetect File Type](#) API endpoint.

**Contents:**

## 3.1 Introduction

Dataverse APIs allow users to accomplish many tasks such as...

- creating datasets
- uploading files
- publishing datasets
- and much, much more

... all without using the Dataverse web interface.

APIs open the door for integrations between Dataverse and other software. For a list, see the *Integrations* section of the Admin Guide.

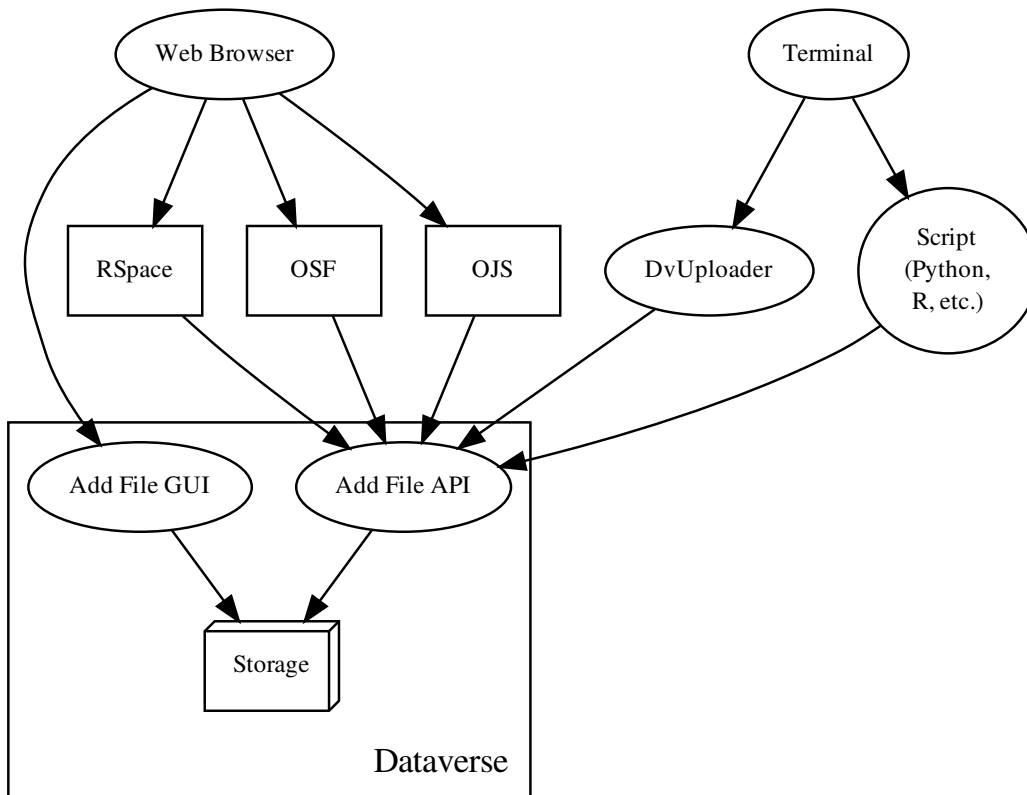
**Contents:**

- *What is an API?*
- *Types of Dataverse API Users*
  - *API Users Within a Single Installation of Dataverse*
    - \* *Users of Integrations and Apps*
    - \* *Power Users*
    - \* *Support Teams and Superusers*
    - \* *Sysadmins*
    - \* *In House Developers*
  - *API Users Across the Dataverse Project*
    - \* *Developers of Integrations, External Tools, and Apps*
    - \* *Developers of Dataverse API Client Libraries*
    - \* *Developers of Dataverse Itself*
- *How This Guide is Organized*
  - *Getting Started*

- *API Tokens and Authentication*
- *Lists of Dataverse APIs*
- *Client Libraries*
- *Examples*
- *Frequently Asked Questions*
- *Getting Help*

### 3.1.1 What is an API?

API stands for “Application Programming Interface” and an example is Dataverse’s “file upload” API. In the diagram below, we can see that while users can click a button within Dataverse’s web interface to upload a file, there are many other ways to get files into Dataverse, all using an API that allows for uploading of files.



The components above that use the “file” upload API are:

- DvUploader is terminal-based application for uploading files that is described in the *Dataset + File Management* section of the User Guide.

- OJS, OSF, and RSpace are all web applications that can integrate with Dataverse and are described in “Getting Data In” in the *Integrations* section of the Admin Guide.
- The script in the diagram can be as simple as a single line of code that is run in a terminal. You can copy and paste “one-liners” like this from the guide. See the *Getting Started with APIs* section for examples using a tool called “curl”.

The diagram above shows only a few examples of software using a specific API but many more APIs are available.

### 3.1.2 Types of Dataverse API Users

This guide is intended to serve multiple audiences but pointers various sections of the guide are provided below based on the type of API user you are.

#### API Users Within a Single Installation of Dataverse

Each installation of Dataverse will have its own groups of people interested in APIs.

#### Users of Integrations and Apps

Integrations and apps can take many forms but two examples are:

- Using Open Science Framework (OSF), a web application, to deposit and publish data into Dataverse.
- Using DVUploader, a terminal-based desktop application, to upload files into Dataverse.

In both examples, users need to obtain an API Token to authenticate with Dataverse.

A good starting point is “API Tokens” in the *Account Creation + Management* section of the User Guide. DvUploader is documented in the *Dataset + File Management* section of the User Guide. The integrations that are enabled depend on your installation of Dataverse. You can find a list in the *Integrations* section of the Admin Guide.

#### Power Users

Power users may be researchers or curators who are comfortable with automating parts of their workflow by writing Python code or similar.

The recommended starting point for power users is the *Getting Started with APIs* section.

#### Support Teams and Superusers

Support teams that answer questions about their installation of Dataverse should familiarize themselves with the *Getting Started with APIs* section to get a sense of common tasks that researchers and curators might be trying to accomplish by using Dataverse APIs.

Superusers of an installation of Dataverse have access a superuser dashboard described in the *Dashboard* section of the Admin Guide but some operations can only be done via API.

A good starting point for both groups is the *Getting Started with APIs* section of this guide followed by the *Troubleshooting* section of the Admin Guide.

### Sysadmins

Sysadmins often write scripts to automate tasks and Dataverse APIs make this possible. Sysadmins have control over the server that Dataverse is running on and may be called upon to execute API commands that are limited to “localhost” (the server itself) for security reasons.

A good starting point for sysadmins is “Blocking API Endpoints” in the *Configuration* section of the Installation Guide, followed by the *Getting Started with APIs* section of this guide, followed by the *Troubleshooting* section of the Admin Guide.

### In House Developers

Some organizations that run Dataverse employ developers who are tasked with using Dataverse APIs to accomplish specific tasks such as building custom integrations with in house systems or creating reports specific to the organization’s needs.

A good starting point for in house developers is the *Getting Started with APIs* section.

### API Users Across the Dataverse Project

The Dataverse project loves contributors! Depending on your interests and skills, you might fall into one or more of the groups below.

#### Developers of Integrations, External Tools, and Apps

One of the primary purposes for Dataverse APIs in the first place is to enable integrations with third party software. Integrations are listed in the following places:

- The *Integrations* section of the Admin Guide.
- The *Building External Tools* section this guide.
- The *Apps* section of this guide.

Good starting points are the three sections above to get a sense of third-party software that already integrates with Dataverse, followed by the *Getting Started with APIs* section.

#### Developers of Dataverse API Client Libraries

A client library helps developers using a specific programming language such as Python, Javascript, R, or Java interact with Dataverse APIs in a manner that is idiomatic for their language. For example, a Python programmer may want to

A good starting point is the *Client Libraries* section, followed by the *Getting Started with APIs* section.

#### Developers of Dataverse Itself

Developers working on Dataverse itself use Dataverse APIs when adding features, fixing bugs, and testing those features and bug fixes.

A good starting point is the *Testing* section of the Developer Guide.

### 3.1.3 How This Guide is Organized

#### Getting Started

See *Getting Started with APIs*

#### API Tokens and Authentication

See *API Tokens and Authentication*.

#### Lists of Dataverse APIs

- *Search API*: For searching dataverses, datasets, and files.
- *Data Access API*: For downloading and subsetting data.
- *Native API*: For performing most tasks that are possible in the GUI. See *Getting Started with APIs* for the most common commands which operate on endpoints with names like:
  - Dataverses
  - Datasets
  - Files
  - etc.
- *Metrics API*: For query statistics about usage of a Dataverse installation.
- *SWORD API*: For depositing data using a standards-based approach rather than the *Native API*.

Please note that some APIs are only documented in other guides that are more suited to their audience:

- Admin Guide
  - *External Tools*
  - *Metadata Customization*
  - *Metadata Export*
  - *Make Data Count*
  - *Geoconnect and WorldMap*
  - *Solr Search Index*
- Installation Guide
  - *Configuration*

#### Client Libraries

See *Client Libraries* for how to use Dataverse APIs from Python, R, and Java.

#### Examples

*Apps* links to example open source code you can study. *Getting Started with APIs* also has many examples.

## Frequently Asked Questions

See *Frequently Asked Questions*.

### 3.1.4 Getting Help

Dataverse API questions are on topic in all the usual places:

- The dataverse-community Google Group: <https://groups.google.com/forum/#!forum/dataverse-community>
- Dataverse community calls: <https://dataverse.org/community-calls>
- The Dataverse chat room: <http://chat.dataverse.org>
- The Dataverse ticketing system: [support@dataverse.org](mailto:support@dataverse.org)

After your question has been answered, you are welcome to help improve the *Frequently Asked Questions* section of this guide.

## 3.2 Getting Started with APIs

If you are a researcher or curator who wants to automate parts of your workflow, this section should help you get started. The *Introduction* section lists resources for other groups who may be interested in Dataverse APIs such as developers of integrations and support teams.

### Contents:

- *Servers You Can Test With*
- *Getting an API Token*
- *curl Examples and Environment Variables*
- *Depositing Data*
  - *Creating a Dataverse*
  - *Creating a Dataset*
  - *Uploading Files*
  - *Publishing a Dataverse*
  - *Publishing a Dataset*
- *Finding and Downloading Data*
  - *Finding Datasets*
  - *Downloading Files*
  - *Downloading Metadata*
  - *Listing the Contents of a Dataverse*
- *Managing Permissions*
  - *Granting Permission*
  - *Revoking Permission*



– *Listing Permissions (Role Assignments)*

- *Beyond “Getting Started” Tasks*
- *Getting Help*

### 3.2.1 Servers You Can Test With

Rather than using a production installation of Dataverse, API users are welcome to use <http://demo.dataverse.org> for testing. You can email [support@dataverse.org](mailto:support@dataverse.org) if you have any trouble with this server.

If you would rather have full control over your own test server, deployments to AWS, Docker, Vagrant, and more are covered in the *Developer Guide* and the *Installation Guide*.

### 3.2.2 Getting an API Token

Many Dataverse APIs require an API token.

Once you have identified a server to test with, create an account, click on your name, and get your API token. For more details, see the *API Tokens and Authentication* section.

### 3.2.3 curl Examples and Environment Variables

The examples in this guide use `curl` for the following reasons:

- `curl` commands are succinct.
- `curl` commands can be copied and pasted into a terminal.
- This guide is programming language agnostic. It doesn’t prefer any particular programming language.

You’ll find `curl` examples that look like this:

```
export SERVER_URL=https://demo.dataverse.org
export QUERY=data

curl $SERVER_URL/api/search?q=$QUERY
```

What’s going on above is the declaration of “environment variables” that are substituted into a `curl` command. You should run the “`export`” commands but change the value for the server URL or the query (or whatever options the command supports). Then you should be able to copy and paste the `curl` command and it should “just work”, substituting the variables like this:

```
curl https://demo.dataverse.org/api/search?q=data
```

If you ever want to check an environment variable, you can “`echo`” it like this:

```
echo $SERVER_URL
```

If you don’t like `curl`, don’t have `curl`, or want to use a different programming language, you are encouraged to check out the Python, Javascript, R, and Java options in the *Client Libraries* section.

## 3.2.4 Depositing Data

### Creating a Dataverse

See *Create a Dataverse*.

### Creating a Dataset

See *Create a Dataset in a Dataverse*.

### Uploading Files

See *Add a File to a Dataset*.

### Publishing a Dataverse

See *Publish a Dataverse*.

### Publishing a Dataset

See *Publish a Dataset*.

## 3.2.5 Finding and Downloading Data

### Finding Datasets

A quick example search for the word “data” is <https://demo.dataverse.org/api/search?q=data>

See the *Search API* section for details.

### Downloading Files

The *Data Access API* section explains how to download files.

In order to download files, you must know their database IDs which you can get from the `dataverse_json` metadata at the dataset level. See *Export Metadata of a Dataset in Various Formats*.

### Downloading Metadata

Dataset metadata is available in a variety of formats listed at *Supported Metadata Export Formats*.

See *Export Metadata of a Dataset in Various Formats*.

### Listing the Contents of a Dataverse

See *Show Contents of a Dataverse*.

## 3.2.6 Managing Permissions

### Granting Permission

See *Assign a New Role on a Dataverse*.

### Revoking Permission

See *Delete Role Assignment from a Dataverse*.

### Listing Permissions (Role Assignments)

See *List Role Assignments in a Dataverse*.

## 3.2.7 Beyond “Getting Started” Tasks

In addition to the tasks listed above, Dataverse supports many other operations via API.

See *Lists of Dataverse APIs* and *Types of Dataverse API Users* to get oriented.

If you’re looking for some inspiration for how you can use Dataverse APIs, there are open source projects that integrate with Dataverse listed in the *Apps* section.

## 3.2.8 Getting Help

See *Getting Help*.

## 3.3 API Tokens and Authentication

An API token is similar to a password and allows you to authenticate to Dataverse APIs to perform actions as you. Many Dataverse APIs require the use of an API token.

### Contents:

- *How to Get an API Token*
- *How Your API Token Is Like a Password*
- *Passing Your API Token as an HTTP Header (Preferred) or a Query Parameter*
- *Resetting Your API Token*

### 3.3.1 How to Get an API Token

Your API token is unique to the server you are using. You cannot take your API token from one server and use it on another server.

Instructions for getting a token are described in the *Account Creation + Management* section of the User Guide.

### 3.3.2 How Your API Token Is Like a Password

Anyone who has your API Token can add and delete data as you so you should treat it with the same care as a password.

### 3.3.3 Passing Your API Token as an HTTP Header (Preferred) or a Query Parameter

See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

There are two ways to pass your API token to Dataverse APIs. The preferred method is to send the token in the `X-Dataverse-key` HTTP header, as in the following `curl` example.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export SERVER_URL=https://demo.dataverse.org
export ALIAS=root

curl -H X-Dataverse-key:$API_TOKEN $SERVER_URL/api/dataverses/$ALIAS/contents
```

Here's how it looks without the environment variables:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx https://demo.dataverse.
↪org/api/dataverses/root/contents
```

The second way to pass your API token is via a query parameter called `key` in the URL like below.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export SERVER_URL=https://demo.dataverse.org
export ALIAS=root

curl $SERVER_URL/api/dataverses/$ALIAS/contents?key=$API_TOKEN
```

Here's how it looks without the environment variables:

```
curl https://demo.dataverse.org/api/dataverses/root/contents?key=xxxxxxxx-xxxx-xxxx-
↪xxxx-xxxxxxxxxxxxx
```

Use of the `X-Dataverse-key` HTTP header form is preferred to passing `key` in the URL because query parameters like `key` appear in URLs and might accidentally get shared, exposing your API token. (Again it's like a password.) Additionally, URLs are often logged on servers while it's less common to log HTTP headers.

### 3.3.4 Resetting Your API Token

You can reset your API Token from your account page in Dataverse as described in the *Account Creation + Management* section of the User Guide.

## 3.4 Search API

**Contents:**

- *Parameters*
- *Basic Search Example*

- [Advanced Search Example](#)
- [Iteration](#)

The Search API supports the same searching, sorting, and faceting operations as the Dataverse web interface.

Unlike the web interface, this new API is limited to *published* data.

The parameters and JSON response are partly inspired by the [GitHub Search API](#).

**Note:** The search API can be used from scripts running in web browsers, as it allows cross-origin resource sharing (CORS).

Please note that in Dataverse 4.3 and older the “citation” field wrapped the persistent ID URL in an `<a>` tag but this has been changed to plaintext. If you want the old value with HTML in it, a new field called “citationHtml” can be used.

### 3.4.1 Parameters

Name	Type	Description
q	string	The search term or terms. Using “title:data” will search only the “title” field. “*” can be used as a wildcard either alone or adjacent to a term (i.e. “bird*”). For example, <a href="https://demo.dataverse.org/api/search?q=title:data">https://demo.dataverse.org/api/search?q=title:data</a> . For a list of fields to search, please see <a href="https://github.com/IQSS/dataverse/issues/2558">https://github.com/IQSS/dataverse/issues/2558</a> (for now).
type	string	Can be either “dataverse”, “dataset”, or “file”. Multiple “type” parameters can be used to include multiple types (i.e. <code>type=dataset&amp;type=file</code> ). If omitted, all types will be returned. For example, <a href="https://demo.dataverse.org/api/search?q=*&amp;type=dataset">https://demo.dataverse.org/api/search?q=*&amp;type=dataset</a>
subtree	string	The identifier of the dataverse to which the search should be narrowed. The subtree of this dataverse and all its children will be searched. Multiple “subtree” parameters can be used to include multiple Dataverses. For example, <a href="https://demo.dataverse.org/api/search?q=data&amp;subtree=birds&amp;subtree=cats">https://demo.dataverse.org/api/search?q=data&amp;subtree=birds&amp;subtree=cats</a> .
sort	string	The sort field. Supported values include “name” and “date”. See example under “order”.
order	string	The order in which to sort. Can either be “asc” or “desc”. For example, <a href="https://demo.dataverse.org/api/search?q=data&amp;sort=name&amp;order=asc">https://demo.dataverse.org/api/search?q=data&amp;sort=name&amp;order=asc</a>
per_page	int	The number of results to return per request. The default is 10. The max is 1000. See <a href="#">iteration example</a> .
start	int	A cursor for paging through search results. See <a href="#">iteration example</a> .
show_relevant_fields	boolean	Whether or not to show details of which fields were matched by the query. False by default. See <a href="#">advanced search example</a> .
show_facets	boolean	Whether or not to show facets that can be operated on by the “fq” parameter. False by default. See <a href="#">advanced search example</a> .
fq	string	A filter query on the search term. Multiple “fq” parameters can be used. See <a href="#">advanced search example</a> .
show_entity_ids	boolean	Whether or not to show the database IDs of the search results (for developer use).
query_entities	boolean	Whether entities are queried via direct database calls (for developer use).

### 3.4.2 Basic Search Example

<https://demo.dataverse.org/api/search?q=trees>

```
{
  "status": "OK",
```

```

"data":{
  "q":"trees",
  "total_count":4,
  "start":0,
  "spelling_alternatives":{
    "trees":["tree"]
  },
  "items":[
    {
      "name":"Trees",
      "type":"dataverse",
      "url":"https://demo.dataverse.org/dataverse/trees",
      "image_url":"https://demo.dataverse.org/api/access/dvCardImage/7",
      "identifier":"trees",
      "description":"A tree dataverse with some birds",
      "published_at":"2016-05-10T12:53:38Z"
    },
    {
      "name":"Chestnut Trees",
      "type":"dataverse",
      "url":"https://demo.dataverse.org/dataverse/chestnuttrees",
      "image_url":"https://demo.dataverse.org/api/access/dvCardImage/9",
      "identifier":"chestnuttrees",
      "description":"A dataverse with chestnut trees and an oriole",
      "published_at":"2016-05-10T12:52:38Z"
    },
    {
      "name":"trees.png",
      "type":"file",
      "url":"https://demo.dataverse.org/api/access/datafile/12",
      "image_url":"https://demo.dataverse.org/api/access/fileCardImage/12",
      "file_id":"12",
      "description":"","",
      "published_at":"2016-05-10T12:53:39Z",
      "file_type":"PNG Image",
      "file_content_type":"image/png",
      "size_in_bytes":8361,
      "md5":"0386269a5acb2c57b4eade587ff4db64",
      "file_persistent_id": "doi:10.5072/FK2/XTT5BV/PCCHV7",
      "dataset_name": "Dataset One",
      "dataset_id": "32",
      "dataset_persistent_id": "doi:10.5072/FK2/XTT5BV",
      "dataset_citation":"Spruce, Sabrina, 2016, \"Spruce Goose\", http://
↪dx.doi.org/10.5072/FK2/XTT5BV, Root Dataverse, V1"
    },
    {
      "name":"Birds",
      "type":"dataverse",
      "url":"https://demo.dataverse.org/dataverse/birds",
      "image_url":"https://demo.dataverse.org/api/access/dvCardImage/2",
      "identifier":"birds",
      "description":"A bird dataverse with some trees",
      "published_at":"2016-05-10T12:57:27Z"
    }
  ],
  "count_in_response":4
}

```

### 3.4.3 Advanced Search Example

[https://demo.dataverse.org/api/search?q=finch&show\\_relevance=true&show\\_facets=true&fq=publicationDate:2016&subtree=birds](https://demo.dataverse.org/api/search?q=finch&show_relevance=true&show_facets=true&fq=publicationDate:2016&subtree=birds)

In this example, `show_relevance=true` matches per field are shown. Available facets are shown with `show_facets=true` and of the facets is being used with `fq=publicationDate:2016`. The search is being narrowed to the dataverse with the identifier “birds” with the parameter `subtree=birds`.

```
{
  "status": "OK",
  "data": {
    "q": "finch",
    "total_count": 2,
    "start": 0,
    "spelling_alternatives": {
    },
    "items": [
      {
        "name": "Finches",
        "type": "dataverse",
        "url": "https://demo.dataverse.org/dataverse/finches",
        "image_url": "https://demo.dataverse.org/api/access/dvCardImage/3",
        "identifier": "finches",
        "description": "A dataverse with finches",
        "published_at": "2016-05-10T12:57:38Z",
        "matches": [
          {
            "description": {
              "snippets": [
                "A dataverse with <span class=\"search-term-match\">
↪ finches</span>"
              ]
            }
          },
          {
            "name": {
              "snippets": [
                "<span class=\"search-term-match\">Finches</span>"
              ]
            }
          }
        ],
        "score": 3.8500118255615234
      },
      {
        "name": "Darwin's Finches",
        "type": "dataset",
        "url": "http://dx.doi.org/10.5072/FK2/G2VPE7",
        "image_url": "https://demo.dataverse.org/api/access/dsCardImage/2",
        "global_id": "doi:10.5072/FK2/G2VPE7",
        "description": "Darwin's finches (also known as the Galápagos_
↪ finches) are a group of about fifteen species of passerine birds.",
        "published_at": "2016-05-10T12:57:45Z",
        "citationHtml": "Finch, Fiona, 2016, \"Darwin's Finches\", <a href=
↪ \"http://dx.doi.org/10.5072/FK2/G2VPE7\" target=\"_blank\">http://dx.doi.org/10.5072/
↪ FK2/G2VPE7</a>, Root Dataverse, V1",
        "citation": "Finch, Fiona, 2016, \"Darwin's Finches\", http://dx.doi.
↪ org/10.5072/FK2/G2VPE7, Root Dataverse, V1",

```

```

        "matches": [
          {
            "authorName": {
              "snippets": [
                "<span class=\"search-term-match\">Finch</span>, Fiona
↵"
              ]
            },
            {
              "dsDescriptionValue": {
                "snippets": [
                  "Darwin's <span class=\"search-term-match\">finches</
↵span> (also known as the Galápagos <span class=\"search-term-match\">finches</span>
↵) are a group of about fifteen species"
                ]
              },
            {
              "title": {
                "snippets": [
                  "Darwin's <span class=\"search-term-match\">Finches</
↵span>"
                ]
              }
            }
          ],
          "score": 1.5033848285675049,
          "authors": [
            "Finch, Fiona"
          ]
        },
        "facets": [
          {
            "subject_ss": {
              "friendly": "Subject",
              "labels": [
                {
                  "Medicine, Health and Life Sciences": 2
                }
              ]
            },
            "authorName_ss": {
              "friendly": "Author Name",
              "labels": [
                {
                  "Finch, Fiona": 1
                }
              ]
            },
            "publicationDate": {
              "friendly": "Publication Date",
              "labels": [
                {
                  "2016": 2
                }
              ]
            }
          ]
        ]
      }
    ]
  }
}

```



```

    }
  ],
  "count_in_response":2
}
}

```

### 3.4.4 Iteration

By default, up to 10 results are returned with every request (though this can be increased with the `per_page` parameter). To iterate through many results, increase the `start` parameter on each iteration until you reach the `total_count` in the response. An example in Python is below.

```

#!/usr/bin/env python
import urllib2
import json
base = 'https://demo.dataverse.org'
rows = 10
start = 0
page = 1
condition = True # emulate do-while
while (condition):
    url = base + '/api/search?q=*' + "&start=" + str(start)
    data = json.load(urllib2.urlopen(url))
    total = data['data']['total_count']
    print "=== Page", page, "==="
    print "start:", start, " total:", total
    for i in data['data']['items']:
        print "- ", i['name'], "(" + i['type'] + ")"
    start = start + rows
    page += 1
    condition = start < total

```

#### Output from iteration example

```

=== Page 1 ===
start: 0 total: 12
- Spruce Goose (dataset)
- trees.png (file)
- Spruce (dataverse)
- Trees (dataverse)
- Darwin's Finches (dataset)
- Finches (dataverse)
- Birds (dataverse)
- Rings of Conifers (dataset)
- Chestnut Trees (dataverse)
- Sparrows (dataverse)
=== Page 2 ===
start: 10 total: 12
- Chestnut Sparrows (dataverse)
- Wrens (dataverse)

```

## 3.5 Data Access API

The Data Access API provides programmatic download access to the files stored under Dataverse. More advanced features of the Access API include format-specific transformations (thumbnail generation/resizing for images; converting tabular data into alternative file formats) and access to the data-level metadata that describes the contents of the tabular files.

### Contents:

- *Basic File Access*
  - *Parameters:*
- *Multiple File (“bundle”) download*
  - *Parameters:*
- *“All Formats” bundled download for Tabular Files.*
  - *Parameters:*
- *Data Variable Metadata Access*
  - *Parameters:*
- *Preprocessed Data*
- *Authentication and Authorization*
- *Access Requests and Processing*
  - *Allow Access Requests:*
  - *Request Access:*
  - *Grant File Access:*
  - *Reject File Access:*
  - *Revoke File Access:*
  - *List File Access Requests:*

### 3.5.1 Basic File Access

Basic access URI:

```
/api/access/datafile/$id
```

---

**Note:** Files can be accessed using persistent identifiers. This is done by passing the constant `:persistentId` where the numeric id of the file is expected, and then passing the actual persistent id as a query parameter with the name `persistentId`.

Example: Getting the file whose DOI is *10.5072/FK2/J8SJZB*

```
GET http://$SERVER/api/access/datafile/:persistentId/?persistentId=doi:10.5072/FK2/
↪ J8SJZB
```

**Parameters:**

format

the following parameter values are supported (for tabular data files only):

Value	Description
original	“Saved Original”, the proprietary (SPSS, Stata, R, etc.) file from which the tabular data was ingested;
RData	Tabular data as an R Data frame (generated; unless the “original” file was in R);
prep	“Pre-processed data”, in JSON.
subset	Column-wise subsetting. You must also supply a comma separated list of variables in the “variables” query parameter. In this example, 123 and 127 are the database ids of data variables that belong to the data file with the id 6: <code>curl 'http://localhost:8080/api/access/datafile/6?format=subset&amp;variables=123,127'</code> .

noVarHeader

(supported for tabular data files only; ignored for all other file types)

Value	Description
true 1	Tab-delimited data file, without the variable name header (added to tab. files by default)

imageThumb

the following parameter values are supported (for image and pdf files only):

Value	Description
true	Generates a thumbnail image, by rescaling to the default thumbnail size (64 pixels)
N	Rescales the image to N pixels.

**3.5.2 Multiple File (“bundle”) download**

```
/api/access/datafiles/$id1,$id2,...$idN
```

Returns the files listed, zipped.

**Note:** If the request can only be completed partially - if only *some* of the requested files can be served (because of the permissions and/or size restrictions), the file MANIFEST.TXT included in the zipped bundle will have entries specifying the reasons the missing files could not be downloaded. IN THE FUTURE the API will return a 207 status code to indicate that the result was a partial success. (As of writing this - v.4.11 - this hasn’t been implemented yet)

**Note:** If any of the datafiles have the `DirectoryLabel` attributes in the corresponding `FileMetadata` entries, these will be added as folders to the Zip archive, and the files will be placed in them accordingly.

**Parameters:**

format the following parameter values are supported (for tabular data files only):

Value	Description
original	“Saved Original”, the proprietary (SPSS, Stata, R, etc.) file from which the tabular data was ingested;

### 3.5.3 “All Formats” bundled download for Tabular Files.

/api/access/datafile/bundle/\$id

This is a convenience packaging method available for tabular data files. It returns a zipped bundle that contains the data in the following formats:

- Tab-delimited;
- “Saved Original”, the proprietary (SPSS, Stata, R, etc.) file from which the tabular data was ingested;
- Generated R Data frame (unless the “original” above was in R);
- Data (Variable) metadata record, in DDI XML;
- File citation, in Endnote and RIS formats.

#### Parameters:

fileMetadataId

Value	Description
ID	Exports file with specific file metadata ID.

### 3.5.4 Data Variable Metadata Access

**These methods are only available for tabular data files. (i.e., data files with associated data table and variable objects).**

/api/access/datafile/\$id/metadata/ddi

In its basic form the verb above returns a DDI fragment that describes the file and the data variables in it.

The DDI returned will only have two top-level sections:

- a single `fileDscr`, with the basic file information plus the numbers of variables and observations and the UNF of the file.
- a single `dataDscr` section, with one `var` section for each variable.

Example:

<http://localhost:8080/api/access/datafile/6/metadata/ddi>

```
<codeBook version="2.0">
  <fileDscr ID="f6">
    <fileTxt>
      <fileName>_73084.tab</fileName>
      <dimensns>
        <caseQnty>3</caseQnty>
        <varQnty>2</varQnty>
      </dimensns>
      <fileType>text/tab-separated-values</fileType>
    </fileTxt>
    <notes level="file" type="VDC:UNF" subject="Universal Numeric Fingerprint">
      ↪UNF:6:zChnyI3fjwNP+6qW0VryVQ==</notes>
    </fileDscr>
    <dataDscr>
      <var ID="v1" name="id" intrvl="discrete">
        <location fileid="f6"/>
        <labl level="variable">Personen-ID</labl>
      </var>
    </dataDscr>
  </fileDscr>
</codeBook>
```

```

    <sumStat type="mean">2.0</sumStat>
    <sumStat type="mode">.</sumStat>
    <sumStat type="medn">2.0</sumStat>
    <sumStat type="stdev">1.0</sumStat>
    <sumStat type="min">1.0</sumStat>
    <sumStat type="vald">3.0</sumStat>
    <sumStat type="invd">0.0</sumStat>
    <sumStat type="max">3.0</sumStat>
    <varFormat type="numeric"/>
    <notes subject="Universal Numeric Fingerprint" level="variable" type="VDC:UNF
↪">UNF:6:AvELPR5QTaBbnq6S22Msow==</notes>
  </var>
  <var ID="v3" name="sex" intrvl="discrete">
    <location fileid="f6"/>
    <labl level="variable">Geschlecht</labl>
    <sumStat type="mean">1.3333333333333333</sumStat>
    <sumStat type="max">2.0</sumStat>
    <sumStat type="vald">3.0</sumStat>
    <sumStat type="mode">.</sumStat>
    <sumStat type="stdev">0.5773502691896257</sumStat>
    <sumStat type="invd">0.0</sumStat>
    <sumStat type="medn">1.0</sumStat>
    <sumStat type="min">1.0</sumStat>
    <catgry>
      <catValu>1</catValu>
      <labl level="category">Mann</labl>
    </catgry>
    <catgry>
      <catValu>2</catValu>
      <labl level="category">Frau</labl>
    </catgry>
    <varFormat type="numeric"/>
    <notes subject="Universal Numeric Fingerprint" level="variable" type="VDC:UNF
↪">UNF:6:XqQaMwOA63taX1YyBzTZYQ==</notes>
  </var>
</dataDscr>
</codeBook>

```

### Parameters:

fileMetadataId

Value	Description
ID	Exports file with specific file metadata ID. For example for data file with id 6 and file metadata id 2: curl 'http://localhost:8080/api/access/datafile/6/metadata/ddi?fileMetadataId=2'

More information on DDI is available in the *Tabular Data, Representation, Storage and Ingest* section of the User Guide.

Advanced options/Parameters:

It is possible to request only specific subsets of, rather than the full file-level DDI record. This can be a useful optimization, in cases such as when an application needs to look up a single variable; especially with data files with large numbers of variables. See `variables=123,127` in the example above.

### 3.5.5 Preprocessed Data

/api/access/datafile/\$id/metadata/preprocessed

This method provides the “preprocessed data” - a summary record that describes the values of the data vectors in the tabular file, in JSON. These metadata values are used by TwoRavens, the companion data exploration utility of the Dataverse application. Please note that this format might change in the future.

### 3.5.6 Authentication and Authorization

Data Access API supports both session- and API key-based authentication.

If a session is available, and it is already associated with an authenticated user, it will be used for access authorization. If not, or if the user in question is not authorized to access the requested object, an attempt will be made to authorize based on an API key, if supplied. All of the API verbs above support the key parameter `key=...` as well as the newer `X-Dataverse-key` header. For more details, see “Authentication” in the *Introduction* section.

### 3.5.7 Access Requests and Processing

All of the following endpoints take the persistent identifier as a parameter in place of ‘id’.

#### Allow Access Requests:

Allow or disallow users from requesting access to restricted files in a dataset where `id` is the database id of the dataset or `pid` is the persistent id (DOI or Handle) of the dataset to update.

A curl example using an `id`:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT -d true http://$SERVER/api/access/{id}/
↳allowAccessRequest
```

A curl example using a `pid`:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT -d true http://$SERVER/api/access/
↳:persistentId/allowAccessRequest?persistentId={pid}
```

#### Request Access:

/api/access/datafile/\$id/requestAccess

This method requests access to the datafile whose `id` is passed on the behalf of an authenticated user whose key is passed. Note that not all datasets allow access requests to restricted files.

A curl example using an `id`:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT http://$SERVER/api/access/datafile/{id}/
↳requestAccess
```

#### Grant File Access:

/api/access/datafile/{id}/grantAccess/{identifier}

This method grants access to the datafile whose id is passed on the behalf of an authenticated user whose user identifier is passed with an @ prefix. The key of a user who can manage permissions of the datafile is required to use this method.

A curl example using an id:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT http://$SERVER/api/access/datafile/{id}/
↳grantAccess/{@userIdentifier}
```

### Reject File Access:

```
/api/access/datafile/{id}/rejectAccess/{identifier}
```

This method rejects the access request to the datafile whose id is passed on the behalf of an authenticated user whose user identifier is passed with an @ prefix. The key of a user who can manage permissions of the datafile is required to use this method.

A curl example using an id:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT http://$SERVER/api/access/datafile/{id}/
↳rejectAccess/{@userIdentifier}
```

### Revoke File Access:

```
/api/access/datafile/{id}/revokeAccess/{identifier}
```

This method revokes previously granted access to the datafile whose id is passed on the behalf of an authenticated user whose user identifier is passed with an @ prefix. The key of a user who can manage permissions of the datafile is required to use this method.

A curl example using an id:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X DELETE http://$SERVER/api/access/datafile/{id}
↳/revokeAccess/{@userIdentifier}
```

### List File Access Requests:

```
/api/access/datafile/{id}/listRequests
```

This method returns a list of Authenticated Users who have requested access to the datafile whose id is passed. The key of a user who can manage permissions of the datafile is required to use this method.

A curl example using an id:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X GET http://$SERVER/api/access/datafile/{id}/
↳listRequests
```

## 3.6 Native API

Dataverse 4 exposes most of its GUI functionality via a REST-based API. This section describes that functionality. Most API endpoints require an API token that can be passed as the X-Dataverse-key HTTP header or in the URL as the key query parameter.

**Note:** Some API endpoint allow **CORS** (cross-origin resource sharing), which makes them usable from scripts running in web browsers. These endpoints are marked with a *CORS* badge.

---

**Note:** Bash environment variables shown below. The idea is that you can “export” these environment variables before copying and pasting the commands that use them. For example, you can set `$SERVER_URL` by running `export SERVER_URL="https://demo.dataverse.org"` in your Bash shell. To check if the environment variable was set properly, you can “echo” it (e.g. `echo $SERVER_URL`). See also [curl Examples and Environment Variables](#).

---

**Warning:** Dataverse 4’s API is versioned at the URI - all API calls may include the version number like so: `http://server-address/api/v1/...`. Omitting the `v1` part would default to the latest API version (currently 1). When writing scripts/applications that will be used for a long time, make sure to specify the API version, so they don’t break when the API is upgraded.

### Contents:

- *Dataverses*
  - *Create a Dataverse*
  - *View a Dataverse*
  - *Delete a Dataverse*
  - *Show Contents of a Dataverse*
  - *Report the data (file) size of a Dataverse*
  - *List Roles Defined in a Dataverse*
  - *List Facets Configured for a Dataverse*
  - *Set Facets for a Dataverse*
  - *Create a New Role in a Dataverse*
  - *List Role Assignments in a Dataverse*
  - *Assign Default Role to User Creating a Dataset in a Dataverse*
  - *Assign a New Role on a Dataverse*
  - *Delete Role Assignment from a Dataverse*
  - *List Metadata Blocks Defined on a Dataverse*
  - *Define Metadata Blocks for a Dataverse*
  - *Determine if a Dataverse Inherits Its Metadata Blocks from Its Parent*
  - *Configure a Dataverse to Inherit Its Metadata Blocks from Its Parent*
  - *Create a Dataset in a Dataverse*
  - *Import a Dataset into a Dataverse*
  - *Import a Dataset into a Dataverse with a DDI file*



- *Publish a Dataverse*
- *Datasets*
  - *Get JSON Representation of a Dataset*
  - *List Versions of a Dataset*
  - *Get Version of a Dataset*
  - *Export Metadata of a Dataset in Various Formats*
    - \* *Schema.org JSON-LD*
  - *List Files in a Dataset*
  - *List All Metadata Blocks for a Dataset*
  - *List Single Metadata Block for a Dataset*
  - *Update Metadata For a Dataset*
  - *Edit Dataset Metadata*
  - *Delete Dataset Metadata*
  - *Publish a Dataset*
  - *Delete Dataset Draft*
  - *Set Citation Date Field for a Dataset*
  - *Revert Citation Date Field to Default for Dataset*
  - *List Role Assignments for a Dataset*
  - *Create a Private URL for a Dataset*
  - *Get the Private URL for a Dataset*
  - *Delete the Private URL from a Dataset*
  - *Add a File to a Dataset*
  - *Submit a Dataset for Review*
  - *Return a Dataset to Author*
  - *Link a Dataset*
  - *Dataset Locks*
  - *Dataset Metrics*
    - \* *Retrieving Total Views for a Dataset*
    - \* *Retrieving Unique Views for a Dataset*
    - \* *Retrieving Total Downloads for a Dataset*
    - \* *Retrieving Unique Downloads for a Dataset*
    - \* *Retrieving Citations for a Dataset*
  - *Delete Unpublished Dataset*
  - *Delete Published Dataset*
- *Files*

- *Adding Files*
- *Accessing (downloading) files*
- *Restrict Files*
- *Uningest a File*
- *Reingest a File*
- *Redetect File Type*
- *Replacing Files*
- *Getting File Metadata*
- *Updating File Metadata*
- *Editing Variable Level Metadata*
- *Provenance*
- *Datafile Integrity*
- *Builtin Users*
  - *Create a Builtin User*
- *Roles*
  - *Create a New Role in a Dataverse*
  - *Show Role*
  - *Delete Role*
- *Explicit Groups*
  - *Create New Explicit Group*
  - *List Explicit Groups in a Dataverse*
  - *Show Single Group in a Dataverse*
  - *Update Group in a Dataverse*
  - *Delete Group from a Dataverse*
  - *Add Multiple Role Assignees to an Explicit Group*
  - *Add a Role Assignee to an Explicit Group*
  - *Remove a Role Assignee from an Explicit Group*
- *Shibboleth Groups*
- *Info*
  - *Show Dataverse Version and Build Number*
  - *Show Dataverse Server Name*
  - *Show Custom Popup Text for Publishing Datasets*
  - *Get API Terms of Use URL*
- *Metadata Blocks*
  - *Show Info About All Metadata Blocks*

- *Show Info About Single Metadata Block*
- *Notifications*
  - *Get All Notifications by User*
- *Admin*
  - *List All Database Settings*
  - *Configure Database Setting*
  - *Get Single Database Setting*
  - *Delete Database Setting*
  - *List Authentication Provider Factories*
  - *List Authentication Providers*
  - *Add Authentication Provider*
  - *Show Authentication Provider*
  - *Enable or Disable an Authentication Provider*
  - *Check If an Authentication Provider is Enabled*
  - *Delete an Authentication Provider*
  - *List Global Roles*
  - *Create Global Role*
  - *List Users*
  - *List Single User*
  - *Merge User Accounts*
  - *Change User Identifier*
  - *Make User a SuperUser*
  - *List Role Assignments of a Role Assignee*
  - *List Permissions a User Has on a Dataverse or Dataset*
  - *Show Role Assignee*
  - *Saved Search*
  - *Dataset Integrity*
  - *Datafile Integrity*
  - *Dataset Validation*
  - *Workflows*
  - *Metrics*
  - *Inherit Dataverse Role Assignments*

### 3.6.1 Dataverses

### Create a Dataverse

A dataverse is a container for datasets and other dataverses as explained in the *Dataverse Management* section of the User Guide.

The steps for creating a dataverse are:

- Prepare a JSON file containing the name, description, etc, of the dataverse you'd like to create.
- Figure out the alias or database id of the “parent” dataverse into which you will be creating your new dataverse.
- Execute a curl command or equivalent.

Download `dataverse-complete.json` file and modify it to suit your needs. The fields `name`, `alias`, and `dataverseContacts` are required. The controlled vocabulary for `dataverseType` is the following:

- DEPARTMENT
- JOURNALS
- LABORATORY
- ORGANIZATIONS\_INSTITUTIONS
- RESEARCHERS
- RESEARCH\_GROUP
- RESEARCH\_PROJECTS
- TEACHING\_COURSES
- UNCATEGORIZED

```
{
  "name": "Scientific Research",
  "alias": "science",
  "dataverseContacts": [
    {
      "contactEmail": "pi@example.edu"
    },
    {
      "contactEmail": "student@example.edu"
    }
  ],
  "affiliation": "Scientific Research University",
  "description": "We do all the science.",
  "dataverseType": "LABORATORY"
}
```

The curl command below assumes you have kept the name “`dataverse-complete.json`” and that this file is in your current working directory.

Next you need to figure out the alias or database id of the “parent” dataverse into which you will be creating your new dataverse. Out of the box the top level dataverse has an alias of “root” and a database id of “1” but your installation may vary. The easiest way to determine the alias of your root dataverse is to click “Advanced Search” and look at the URL. You may also choose a parent under the root.

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export PARENT=root
export SERVER_URL=https://demo.dataverse.org

curl -H X-Dataverse-key:$API_TOKEN -X POST $SERVER_URL/api/dataverses/$PARENT --
↳upload-file dataverse-complete.json
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx -X POST https://demo.
↳dataverse.org/api/dataverses/root --upload-file dataverse-complete.json
```

You should expect a 201 (“CREATED”) response and JSON indicating the database id that has been assigned to your newly created dataverse.

### View a Dataverse

View data about the dataverse identified by `$id`. `$id` can be the id number of the dataverse, its identifier (a.k.a. alias), or the special value `:root` for the root dataverse.

```
curl $SERVER_URL/api/dataverses/$id
```

### Delete a Dataverse

In order to delete a dataverse you must first delete or move all of its contents elsewhere.

Deletes the dataverse whose ID is given:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X DELETE $SERVER_URL/api/dataverses/$id
```

### Show Contents of a Dataverse

Lists all the dataverses and datasets directly under a dataverse (direct children only). You must specify the “alias” of a dataverse or its database id. If you specify your API token and have access, unpublished dataverses and datasets will be included in the listing.

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export ALIAS=root
export SERVER_URL=https://demo.dataverse.org

curl -H X-Dataverse-key:$API_TOKEN $SERVER_URL/api/dataverses/$ALIAS/contents
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx https://demo.dataverse.
↳org/api/dataverses/root/contents
```

### Report the data (file) size of a Dataverse

Shows the combined size in bytes of all the files uploaded into the dataverse `id`.

```
``curl -H "X-Dataverse-key:$API_TOKEN" http://$SERVER_URL/api/dataverses/$id/  
↪storageSize``
```

Both published and unpublished files will be counted, in the dataverse specified, and in all its sub-dataverses, recursively. By default, only the archival files are counted - i.e., the files uploaded by users (plus the tab-delimited versions generated for tabular data files on ingest). If the optional argument `includeCached=true` is specified, the API will also add the sizes of all the extra files generated and cached by Dataverse - the resized thumbnail versions for image files, the metadata exports for published datasets, etc.

### List Roles Defined in a Dataverse

All the roles defined directly in the dataverse identified by `id`:

```
GET http://$SERVER/api/dataverses/$id/roles?key=$apiKey
```

### List Facets Configured for a Dataverse

List all the facets for a given dataverse `id`.

```
GET http://$SERVER/api/dataverses/$id/facets?key=$apiKey
```

### Set Facets for a Dataverse

Assign search facets for a given dataverse with alias `$alias`

```
curl -H "X-Dataverse-key: $apiKey" -X POST http://$server/api/dataverses/  
$alias/facets --upload-file facets.json
```

Where `facets.json` contains a JSON encoded list of metadata keys (e.g. `["authorName", "authorAffiliation"]`).

### Create a New Role in a Dataverse

Creates a new role under dataverse `id`. Needs a json file with the role description:

```
POST http://$SERVER/api/dataverses/$id/roles?key=$apiKey
```

POSTed JSON example:

```
{  
  "alias": "sys1",  
  "name": "Restricted System Role",  
  "description": "A person who may only add datasets.",  
  "permissions": [  
    "AddDataset"  
  ]  
}
```

### List Role Assignments in a Dataverse

List all the role assignments at the given dataverse:

```
GET http://$SERVER/api/dataverses/$id/assignments?key=$apiKey
```

### Assign Default Role to User Creating a Dataset in a Dataverse

Assign a default role to a user creating a dataset in a dataverse `id` where `roleAlias` is the database alias of the role to be assigned:

```
PUT http://$SERVER/api/dataverses/$id/defaultContributorRole/$roleAlias?key=$apiKey
```

Note: You may use “none” as the `roleAlias`. This will prevent a user who creates a dataset from having any role on that dataset. It is not recommended for dataverses with human contributors.

### Assign a New Role on a Dataverse

Assigns a new role, based on the POSTed JSON.

```
POST http://$SERVER/api/dataverses/$id/assignments?key=$apiKey
```

POSTed JSON example:

```
{
  "assignee": "@uma",
  "role": "curator"
}
```

### Delete Role Assignment from a Dataverse

Delete the assignment whose id is `$id`:

```
DELETE http://$SERVER/api/dataverses/$id/assignments/$id?key=$apiKey
```

### List Metadata Blocks Defined on a Dataverse

Get the metadata blocks defined on a dataverse which determine which field are available to authors when they create and edit datasets within that dataverse. This feature is described under “General Information” in the *Dataverse Management* section of the User Guide.

Please note that an API token is only required if the dataverse has not been published.

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
export ALIAS=root
export SERVER_URL=https://demo.dataverse.org
curl -H X-Dataverse-key:$API_TOKEN $SERVER_URL/api/dataverses/$ALIAS/metadatablocks
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx https://demo.dataverse.org/api/dataverses/root/metadatablocks
```

### Define Metadata Blocks for a Dataverse

You can define the metadata blocks available to authors within a dataverse.

The metadata blocks that are available with a default installation of Dataverse are in `define-metadatablocks.json` (also shown below) and you should download this file and edit it to meet your needs. Please note that the “citation” metadata block is required. You must have “EditDataverse” permission on the dataverse.

```
[
  "citation",
  "geospatial",
  "socialscience",
  "astrophysics",
  "biomedical",
  "journal"
]
```

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export ALIAS=root
export SERVER_URL=https://demo.dataverse.org

curl -H X-Dataverse-key:$API_TOKEN -X POST -H \"Content-type:application/json\" --
  ↪upload-file define-metadatablocks.json $SERVER_URL/api/dataverses/$ALIAS/
  ↪metadatablocks
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx -X POST -H \"Content-
  ↪type:application/json\" --upload-file define-metadatablocks.json https://demo.
  ↪dataverse.org/api/dataverses/root/metadatablocks
```

### Determine if a Dataverse Inherits Its Metadata Blocks from Its Parent

Get whether the dataverse is a metadata block root, or does it uses its parent blocks:

```
GET http://$SERVER/api/dataverses/$id/metadatablocks/isRoot?key=$apiKey
```

### Configure a Dataverse to Inherit Its Metadata Blocks from Its Parent

Set whether the dataverse is a metadata block root, or does it uses its parent blocks. Possible values are `true` and `false` (both are valid JSON expressions).

```
PUT http://$SERVER/api/dataverses/$id/metadatablocks/isRoot?key=$apiKey
```



---

**Note:** Previous endpoints `GET http://$SERVER/api/dataverses/$id/metadatablocks/:isRoot?key=$apiKey` and `POST http://$SERVER/api/dataverses/$id/metadatablocks/:isRoot?key=$apiKey` are deprecated, but supported.

---

## Create a Dataset in a Dataverse

A dataset is a container for files as explained in the *Dataset + File Management* section of the User Guide.

To create a dataset, you must supply a JSON file that contains at least the following required metadata fields:

- Title
- Author
- Description
- Subject

As a starting point, you can download `dataset-finch1.json` and modify it to meet your needs. (In addition to this minimal example, you can download `dataset-create-new-all-default-fields.json` which populates all of the metadata fields that ship with Dataverse.)

The curl command below assumes you have kept the name “dataset-finch1.json” and that this file is in your current working directory.

Next you need to figure out the alias or database id of the “parent” dataverse into which you will be creating your new dataset. Out of the box the top level dataverse has an alias of “root” and a database id of “1” but your installation may vary. The easiest way to determine the alias of your root dataverse is to click “Advanced Search” and look at the URL. You may also choose a parent dataverse under the root dataverse.

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export PARENT=root
export SERVER_URL=https://demo.dataverse.org

curl -H X-Dataverse-key:$API_TOKEN -X POST $SERVER_URL/api/dataverses/$PARENT/
↳datasets --upload-file dataset-finch1.json
```

The fully expanded example above (without the environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx -X POST https://demo.
↳dataverse.org/api/dataverses/root/datasets --upload-file dataset-finch1.json
```

You should expect a 201 (“CREATED”) response and JSON indicating the database ID and Persistent ID (PID such as DOI or Handle) that has been assigned to your newly created dataset.

## Import a Dataset into a Dataverse

---

**Note:** This action requires a Dataverse account with super-user permissions.

---

To import a dataset with an existing persistent identifier (PID), the dataset's metadata should be prepared in Dataverse's native JSON format. The PID is provided as a parameter at the URL. The following line imports a dataset with the PID `PERSISTENT_IDENTIFIER` to Dataverse, and then releases it:

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST $SERVER_URL/api/dataverses/$DV_ALIAS/  
↳datasets/:import?pid=$PERSISTENT_IDENTIFIER&release=yes --upload-file dataset.json
```

The `pid` parameter holds a persistent identifier (such as a DOI or Handle). The import will fail if no PID is provided, or if the provided PID fails validation.

The optional `release` parameter tells Dataverse to immediately publish the dataset. If the parameter is changed to `no`, the imported dataset will remain in DRAFT status.

The JSON format is the same as that supported by the native API's *create dataset command*, although it also allows packages. For example:

```
{  
  "datasetVersion": {  
    "termsOfUse": "CC0 Waiver",  
    "license": "CC0",  
    "protocol": "doi",  
    "authority": "10.502",  
    "identifier": "ZZ7/MOSEISLEYDB94",  
    "metadataBlocks": {  
      "citation": {  
        "fields": [  
          {  
            "typeName": "title",  
            "multiple": false,  
            "value": "Imported dataset with package files No. 3",  
            "typeClass": "primitive"  
          },  
          {  
            "typeName": "productionDate",  
            "multiple": false,  
            "value": "2011-02-23",  
            "typeClass": "primitive"  
          },  
          {  
            "typeName": "dsDescription",  
            "multiple": true,  
            "value": [  
              {  
                "dsDescriptionValue": {  
                  "typeName": "dsDescriptionValue",  
                  "multiple": false,  
                  "value": "Native Dataset",  
                  "typeClass": "primitive"  
                }  
              }  
            ],  
            "typeClass": "compound"  
          },  
          {  
            "typeName": "subject",  
            "multiple": true,  
            "value": [  
              "Medicine, Health and Life Sciences"  
            ],  
            "typeClass": "compound"  
          }  
        ]  
      }  
    }  
  }  
}
```

```

    "typeClass": "controlledVocabulary"
  },
  {
    "typeName": "author",
    "multiple": true,
    "value": [
      {
        "authorAffiliation": {
          "typeName": "authorAffiliation",
          "multiple": false,
          "value": "LibraScholar Medical School",
          "typeClass": "primitive"
        },
        "authorName": {
          "typeName": "authorName",
          "multiple": false,
          "value": "Doc, Bob",
          "typeClass": "primitive"
        }
      },
      {
        "authorAffiliation": {
          "typeName": "authorAffiliation",
          "multiple": false,
          "value": "LibraScholar Medical School",
          "typeClass": "primitive"
        },
        "authorName": {
          "typeName": "authorName",
          "multiple": false,
          "value": "Prof, Arthur",
          "typeClass": "primitive"
        }
      }
    ],
    "typeClass": "compound"
  },
  {
    "typeName": "depositor",
    "multiple": false,
    "value": "Prof, Arthur",
    "typeClass": "primitive"
  },
  {
    "typeName": "datasetContact",
    "multiple": true,
    "value": [
      {
        "datasetContactEmail": {
          "typeName": "datasetContactEmail",
          "multiple": false,
          "value": "aprof@mailinator.com",
          "typeClass": "primitive"
        }
      }
    ],
    "typeClass": "compound"
  }
}

```

```
    ],
    "displayName": "Citation Metadata"
  }
},
"files": [
  {
    "description": "",
    "label": "pub",
    "restricted": false,
    "version": 1,
    "datasetVersionId": 1,
    "dataFile": {
      "id": 4,
      "filename": "pub",
      "contentType": "application/vnd.dataverse.file-package",
      "filesize": 1698795873,
      "description": "",
      "storageIdentifier": "162017e5ad5-ee2a2b17fee9",
      "originalFormatLabel": "UNKNOWN",
      "rootDataFileId": -1,
      "checksum": {
        "type": "SHA-1",
        "value": "54bc7ddb096a490474bd8cc90cbcd1c96730f350"
      }
    }
  }
]
}
```

Before calling the API, make sure the data files referenced by the POSTed JSON are placed in the dataset directory with filenames matching their specified storage identifiers. In installations using POSIX storage, these files must be made readable by GlassFish.

---

**Tip:** If possible, it's best to avoid spaces and special characters in the storage identifier in order to avoid potential portability problems. The storage identifier corresponds with the filesystem name (or bucket identifier) of the data file, so these characters may cause unpredictability with filesystem tools.

---

**Warning:**

- This API does not cover staging files (with correct contents, checksums, sizes, etc.) in the corresponding places in the Dataverse filestore.
- This API endpoint does not support importing *files'* persistent identifiers.
- A Dataverse server can import datasets with a valid PID that uses a different protocol or authority than said server is configured for. However, the server will not update the PID metadata on subsequent update and publish actions.

### Import a Dataset into a Dataverse with a DDI file

---

**Note:** This action requires a Dataverse account with super-user permissions.

---

To import a dataset with an existing persistent identifier (PID), you have to provide the PID as a parameter at the URL. The following line imports a dataset with the PID `PERSISTENT_IDENTIFIER` to Dataverse, and then releases it:

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST $SERVER_URL/api/dataverses/$DV_ALIAS/
↳datasets/:importddi?pid=$PERSISTENT_IDENTIFIER&release=yes --upload-file ddi_
↳dataset.xml
```

The optional `pid` parameter holds a persistent identifier (such as a DOI or Handle). The import will fail if the provided PID fails validation.

The optional `release` parameter tells Dataverse to immediately publish the dataset. If the parameter is changed to `no`, the imported dataset will remain in `DRAFT` status.

The file is a DDI xml file.

#### Warning:

- This API does not handle files related to the DDI file.
- A Dataverse server can import datasets with a valid PID that uses a different protocol or authority than said server is configured for. However, the server will not update the PID metadata on subsequent update and publish actions.

## Publish a Dataverse

In order to publish a dataverse, you must know either its “alias” (which the GUI calls an “identifier”) or its database ID.

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export ALIAS=root
export SERVER_URL=https://demo.dataverse.org

curl -H X-Dataverse-key:$API_TOKEN -X POST $SERVER_URL/api/dataverses/$ALIAS/actions/
↳:publish
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx -X POST https://demo.
↳dataverse.org/api/dataverses/root/actions/:publish
```

You should expect a 200 (“OK”) response and JSON output.

## 3.6.2 Datasets

**Note** Creation of new datasets is done with a `POST` onto dataverses. See *Dataverses* section.

**Note** In all commands below, dataset versions can be referred to as:

- `:draft` the draft version, if any
- `:latest` either a draft (if exists) or the latest published version.
- `:latest-published` the latest published version

- `x.y` a specific version, where `x` is the major version number and `y` is the minor version number.
- `x` same as `x.0`

### Get JSON Representation of a Dataset

**Note:** Datasets can be accessed using persistent identifiers. This is done by passing the constant `:persistentId` where the numeric id of the dataset is expected, and then passing the actual persistent id as a query parameter with the name `persistentId`.

Example: Getting the dataset whose DOI is *10.5072/FK2/J8SJZB*

```
curl http://$SERVER/api/datasets/:persistentId/?persistentId=doi:10.5072/FK2/J8SJZB
```

fully expanded:

```
curl http://localhost:8080/api/datasets/:persistentId/?persistentId=doi:10.5072/FK2/J8SJZB
```

Getting its draft version:

```
curl http://$SERVER/api/datasets/:persistentId/versions/:draft?persistentId=doi:10.5072/FK2/J8SJZB
```

fully expanded:

```
curl http://localhost:8080/api/datasets/:persistentId/versions/:draft?persistentId=doi:10.5072/FK2/J8SJZB
```

Show the dataset whose id is passed:

```
GET http://$SERVER/api/datasets/$id?key=$apiKey
```

### List Versions of a Dataset

List versions of the dataset:

```
GET http://$SERVER/api/datasets/$id/versions?key=$apiKey
```

### Get Version of a Dataset

Show a version of the dataset. The Dataset also include any metadata blocks the data might have:

```
GET http://$SERVER/api/datasets/$id/versions/$versionNumber?key=$apiKey
```

### Export Metadata of a Dataset in Various Formats

Export the metadata of the current published version of a dataset in various formats see Note below:

```
GET http://$SERVER/api/datasets/export?exporter=ddi&persistentId=$persistentId
```

---

**Note:** Supported exporters (export formats) are `ddi`, `oai_ddi`, `dcterms`, `oai_dc`, `schema.org`, `OAI_ORE`, `Datacite`, `oai_datacite` and `dataverse_json`. Descriptive names can be found under *Supported Metadata Export Formats* in the User Guide.

---

## Schema.org JSON-LD

Please note that the `schema.org` format has changed in backwards-incompatible ways after Dataverse 4.9.4:

- “description” was a single string and now it is an array of strings.
- “citation” was an array of strings and now it is an array of objects.

Both forms are valid according to Google’s Structured Data Testing Tool at <https://search.google.com/structured-data/testing-tool>. (This tool will report “The property affiliation is not recognized by Google for an object of type Thing” and this known issue is being tracked at <https://github.com/IQSS/dataverse/issues/5029>.) Schema.org JSON-LD is an evolving standard that permits a great deal of flexibility. For example, [https://schema.org/docs/gs.html#schemaorg\\_expected](https://schema.org/docs/gs.html#schemaorg_expected) indicates that even when objects are expected, it’s ok to just use text. As with all metadata export formats, we will try to keep the Schema.org JSON-LD format Dataverse emits backward-compatible to made integrations more stable, despite the flexibility that’s afforded by the standard.

## List Files in a Dataset

Lists all the file metadata, for the given dataset and version:

```
GET http://$SERVER/api/datasets/$id/versions/$versionId/files?key=$apiKey
```

## List All Metadata Blocks for a Dataset

Lists all the metadata blocks and their content, for the given dataset and version:

```
GET http://$SERVER/api/datasets/$id/versions/$versionId/metadata?key=$apiKey
```

## List Single Metadata Block for a Dataset

Lists the metadata block named *blockname*, for the given dataset and version:

```
GET http://$SERVER/api/datasets/$id/versions/$versionId/metadata/$blockname?key=
↪$apiKey
```

## Update Metadata For a Dataset

Updates the metadata for a dataset. If a draft of the dataset already exists, the metadata of that draft is overwritten; otherwise, a new draft is created with this metadata.

You must download a JSON representation of the dataset, edit the JSON you download, and then send the updated JSON to the Dataverse server.

For example, after making your edits, your JSON file might look like `dataset-update-metadata.json` which you would send to Dataverse like this:

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT $SERVER_URL/api/datasets/:persistentId/  
↳versions/:draft?persistentId=$PID --upload-file dataset-update-metadata.json
```

Note that in the example JSON file above, there is a single JSON object with `metadataBlocks` as a key. When you download a representation of your dataset in JSON format, the `metadataBlocks` object you need is nested inside another object called `json`. To extract just the `metadataBlocks` key when downloading a JSON representation, you can use a tool such as `jq` like this:

```
curl -H "X-Dataverse-key: $API_TOKEN" $SERVER_URL/api/datasets/:persistentId/versions/  
↳:latest?persistentId=$PID | jq '.data | {metadataBlocks: .metadataBlocks}' >_  
↳dataset-update-metadata.json
```

Now that the resulting JSON file only contains the `metadataBlocks` key, you can edit the JSON such as with `vi` in the example below:

```
vi dataset-update-metadata.json
```

Now that you've made edits to the metadata in your JSON file, you can send it to Dataverse as described above.

### Edit Dataset Metadata

Alternatively to replacing an entire dataset version with its JSON representation you may add data to dataset fields that are blank or accept multiple values with the following

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT $SERVER_URL/api/datasets/:persistentId/  
↳editMetadata/?persistentId=$PID --upload-file dataset-add-metadata.json
```

You may also replace existing metadata in dataset fields with the following (adding the parameter `replace=true`)

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT $SERVER_URL/api/datasets/:persistentId/  
↳editMetadata?persistentId=$PID&replace=true --upload-file dataset-update-metadata.  
↳json
```

For these edits your JSON file need only include those dataset fields which you would like to edit. A sample JSON file may be downloaded here: `dataset-edit-metadata-sample.json`

### Delete Dataset Metadata

You may delete some of the metadata of a dataset version by supplying a file with a JSON representation of dataset fields that you would like to delete with the following

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT $SERVER_URL/api/datasets/:persistentId/  
↳deleteMetadata/?persistentId=$PID --upload-file dataset-delete-author-metadata.json
```

For these deletes your JSON file must include an exact match of those dataset fields which you would like to delete. A sample JSON file may be downloaded here: `dataset-delete-author-metadata.json`

### Publish a Dataset

When publishing a dataset it's good to be aware of Dataverse's versioning system, which is described in the [Dataset + File Management](#) section of the User Guide.

If this is the first version of the dataset, its version number will be set to 1.0. Otherwise, the new dataset version number is determined by the most recent version number and the `type` parameter. Passing `type=minor` increases



the minor version number (2.3 is updated to 2.4). Passing `type=major` increases the major version number (2.3 is updated to 3.0). (Superusers can pass `type=updatecurrent` to update metadata without changing the version number.)

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export SERVER_URL=https://demo.dataverse.org
export PERSISTENT_ID=doi:10.5072/FK2/J8SJZB
export MAJOR_OR_MINOR=major

curl -H X-Dataverse-key:$API_TOKEN -X POST \
"$SERVER_URL/api/datasets/:persistentId/
↪actions/:publish?persistentId=$PERSISTENT_ID&type=$MAJOR_OR_MINOR\""
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx -X POST "https://demo.
↪dataverse.org/api/datasets/:persistentId/actions/:publish?persistentId=doi:10.5072/
↪FK2/J8SJZB&type=major"
```

The quotes around the URL are required because there is more than one query parameter separated by an ampersand (&), which has special meaning to Unix shells such as Bash. Putting the & in quotes ensures that “type” is interpreted as one of the query parameters.

You should expect JSON output and a 200 (“OK”) response in most cases. If you receive a 202 (“ACCEPTED”) response, this is normal for installations that have workflows configured. Workflows are described in the *Workflows* section of the Developer Guide.

---

**Note:** POST should be used to publish a dataset. GET is supported for backward compatibility but is deprecated and may be removed: <https://github.com/IQSS/dataverse/issues/2431>

---

### Delete Dataset Draft

Deletes the draft version of dataset `$id`. Only the draft version can be deleted:

```
DELETE http://$SERVER/api/datasets/$id/versions/:draft?key=$apiKey
```

### Set Citation Date Field for a Dataset

Sets the dataset field type to be used as the citation date for the given dataset (if the dataset does not include the dataset field type, the default logic is used). The name of the dataset field type should be sent in the body of the request. To revert to the default logic, use `:publicationDate` as the `$datasetFieldName`. Note that the dataset field used has to be a date field:

```
PUT http://$SERVER/api/datasets/$id/citationdate?key=$apiKey --data "
↪$datasetFieldName"
```

### Revert Citation Date Field to Default for Dataset

Restores the default logic of the field type to be used as the citation date. Same as PUT with `:publicationDate` body:

```
DELETE http://$SERVER/api/datasets/$id/citationdate?key=$apiKey
```

### List Role Assignments for a Dataset

List all the role assignments at the given dataset:

```
GET http://$SERVER/api/datasets/$id/assignments?key=$apiKey
```

### Create a Private URL for a Dataset

Create a Private URL (must be able to manage dataset permissions):

```
POST http://$SERVER/api/datasets/$id/privateUrl?key=$apiKey
```

### Get the Private URL for a Dataset

Get a Private URL from a dataset (if available):

```
GET http://$SERVER/api/datasets/$id/privateUrl?key=$apiKey
```

### Delete the Private URL from a Dataset

Delete a Private URL from a dataset (if it exists):

```
DELETE http://$SERVER/api/datasets/$id/privateUrl?key=$apiKey
```

### Add a File to a Dataset

When adding a file to a dataset, you can optionally specify the following:

- A description of the file.
- The “File Path” of the file, indicating which folder the file should be uploaded to within the dataset.
- Whether or not the file is restricted.

In the curl example below, all of the above are specified but they are optional.

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
export FILENAME='data.tsv'
export SERVER_URL=https://demo.dataverse.org
export PERSISTENT_ID=doi:10.5072/FK2/J8SJZB

curl -H X-Dataverse-key:$API_TOKEN -X POST -F "file=@$FILENAME" -F 'jsonData={
↪ "description": "My description.", "directoryLabel": "data/subdir1", "categories": ["Data
↪"], "restrict": "false"}' "$SERVER_URL/api/datasets/:persistentId/add?persistentId=
↪ $PERSISTENT_ID"
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -X POST -F file=@data.
↪tsv -F jsonData={"description":"My description.", "directoryLabel":"data/subdir1",
↪"categories":["Data"], "restrict":"false"} https://demo.dataverse.org/api/datasets/
↪:persistentId/add?persistentId=doi:10.5072/FK2/J8SJZB
```

You should expect a 201 (“CREATED”) response and JSON indicating the database id that has been assigned to your newly uploaded file.

Please note that it’s possible to “trick” Dataverse into giving a file a content type (MIME type) of your choosing. For example, you can make a text file be treated like a video file with `-F 'file=@README.txt;type=video/mpeg4'`, for example. If Dataverse does not properly detect a file type, specifying the content type via API like this a potential workaround.

The curl syntax above to upload a file is tricky and a Python version is provided below. (Please note that it depends on libraries such as “requests” that you may need to install but this task is out of scope for this guide.) Here are some parameters you can set in the script:

- `dataverse_server` - e.g. <https://demo.dataverse.org>
- `api_key` - See the top of this document for a description
- `persistentId` - Example: `doi:10.5072/FK2/6XACVA`
- `dataset_id` - Database id of the dataset

In practice, you only need one the `dataset_id` or the `persistentId`. The example below shows both uses.

```
from datetime import datetime
import json
import requests # http://docs.python-requests.org/en/master/

# -----
# Update the 4 params below to run this code
# -----
dataverse_server = 'https://your dataverse server' # no trailing slash
api_key = 'api key'
dataset_id = 1 # database id of the dataset
persistentId = 'doi:10.5072/FK2/6XACVA' # doi or hdl of the dataset

# -----
# Prepare "file"
# -----
file_content = 'content: %s' % datetime.now()
files = {'file': ('sample_file.txt', file_content)}

# -----
# Using a "jsonData" parameter, add optional description + file tags
# -----
params = dict(description='Blue skies!',
              categories=['Lily', 'Rosemary', 'Jack of Hearts'])

params_as_json_string = json.dumps(params)

payload = dict(jsonData=params_as_json_string)

# -----
# Add file using the Dataset's id
# -----
url_dataset_id = '%s/api/datasets/%s/add?key=%s' % (dataverse_server, dataset_id, api_
↪key)
```

```

# -----
# Make the request
# -----
print '-' * 40
print 'making request: %s' % url_dataset_id
r = requests.post(url_dataset_id, data=payload, files=files)

# -----
# Print the response
# -----
print '-' * 40
print r.json()
print r.status_code

# -----
# Add file using the Dataset's persistentId (e.g. doi, hdl, etc)
# -----
url_persistent_id = '%s/api/datasets/:persistentId/add?persistentId=%s&key=%s' % (
↳(dataverse_server, persistentId, api_key)

# -----
# Update the file content to avoid a duplicate file error
# -----
file_content = 'content2: %s' % datetime.now()
files = {'file': ('sample_file2.txt', file_content)}

# -----
# Make the request
# -----
print '-' * 40
print 'making request: %s' % url_persistent_id
r = requests.post(url_persistent_id, data=payload, files=files)

# -----
# Print the response
# -----
print '-' * 40
print r.json()
print r.status_code

```

### Submit a Dataset for Review

When dataset authors do not have permission to publish directly, they can click the “Submit for Review” button in the web interface (see *Dataset + File Management*), or perform the equivalent operation via API:

```

curl -H "X-Dataverse-key: $API_TOKEN" -X POST "$SERVER_URL/api/datasets/:persistentId/
↳submitForReview?persistentId=$DOI_OR_HANDLE_OF_DATASET"

```

The people who need to review the dataset (often curators or journal editors) can check their notifications periodically via API to see if any new datasets have been submitted for review and need their attention. See the *Notifications* section for details. Alternatively, these curators can simply check their email or notifications to know when datasets have been submitted (or resubmitted) for review.

## Return a Dataset to Author

After the curators or journal editors have reviewed a dataset that has been submitted for review (see “Submit for Review”, above) they can either choose to publish the dataset (see the `:publish` “action” above) or return the dataset to its authors. In the web interface there is a “Return to Author” button (see *Dataset + File Management*), but the interface does not provide a way to explain **why** the dataset is being returned. There is a way to do this outside of this interface, however. Instead of clicking the “Return to Author” button in the UI, a curator can write a “reason for return” into the database via API.

Here’s how curators can send a “reason for return” to the dataset authors. First, the curator creates a JSON file that contains the reason for return:

```
{
  "reasonForReturn": "You forgot to upload any files."
}
```

In the example below, the curator has saved the JSON file as `reason-for-return.json` in their current working directory. Then, the curator sends this JSON file to the `returnToAuthor` API endpoint like this:

```
curl -H "Content-type:application/json" -d @reason-for-return.json -H "X-Dataverse-
↪key: $API_TOKEN" -X POST "$SERVER_URL/api/datasets/:persistentId/returnToAuthor?
↪persistentId=$DOI_OR_HANDLE_OF_DATASET"
```

The review process can sometimes resemble a tennis match, with the authors submitting and resubmitting the dataset over and over until the curators are satisfied. Each time the curators send a “reason for return” via API, that reason is persisted into the database, stored at the dataset version level.

## Link a Dataset

Creates a link between a dataset and a dataverse (see the *Linked Dataverses + Linked Datasets* section of the *Dataverse Management* guide for more information).

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT http://$SERVER/api/datasets/$linked-
↪dataset-id/link/$linking-dataverse-alias
```

## Dataset Locks

To check if a dataset is locked:

```
curl "$SERVER_URL/api/datasets/{database_id}/locks
```

Optionally, you can check if there’s a lock of a specific type on the dataset:

```
curl "$SERVER_URL/api/datasets/{database_id}/locks?type={lock_type}
```

Currently implemented lock types are `Ingest`, `Workflow`, `InReview`, `DcmUpload`, `pidRegister`, and `EditInProgress`.

The API will output the list of locks, for example:

```
{ "status": "OK", "data":
  [
    {
      "lockType": "Ingest",
      "date": "Fri Aug 17 15:05:51 EDT 2018",
```

```
    "user": "dataverseAdmin"
  },
  {
    "lockType": "Workflow",
    "date": "Fri Aug 17 15:02:00 EDT 2018",
    "user": "dataverseAdmin"
  }
]
}
```

If the dataset is not locked (or if there is no lock of the requested type), the API will return an empty list.

The following API end point will lock a Dataset with a lock of specified type:

```
POST /api/datasets/{database_id}/lock/{lock_type}
```

For example:

```
curl -X POST "$SERVER_URL/api/datasets/1234/lock/Ingest?key=$ADMIN_API_TOKEN"
or
curl -X POST -H "X-Dataverse-key: $ADMIN_API_TOKEN" "$SERVER_URL/api/datasets/
↳:persistentId/lock/Ingest?persistentId=$DOI_OR_HANDLE_OF_DATASET"
```

Use the following API to unlock the dataset, by deleting all the locks currently on the dataset:

```
DELETE /api/datasets/{database_id}/locks
```

Or, to delete a lock of the type specified only:

```
DELETE /api/datasets/{database_id}/locks?type={lock_type}
```

For example:

```
curl -X DELETE -H "X-Dataverse-key: $ADMIN_API_TOKEN" "$SERVER_URL/api/datasets/1234/
↳locks?type=pidRegister"
```

If the dataset is not locked (or if there is no lock of the specified type), the API will exit with a warning message.

(Note that the API calls above all support both the database id and persistent identifier notation for referencing the dataset)

## Dataset Metrics

Please note that these dataset level metrics are only available if support for Make Data Count has been enabled in your installation of Dataverse. See the *Dataset Metrics* in the *Dataset + File Management* section of the User Guide and the *Make Data Count* section of the Admin Guide for details.

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of `export` below.

---

```
export DV_BASE_URL=https://demo.dataverse.org
```

To confirm that the environment variable was set properly, you can use `echo` like this:

```
echo $DV_BASE_URL
```

Please note that for each of these endpoints except the “citations” endpoint, you can optionally pass the query parameter “country” with a two letter code (e.g. “country=us”) and you can specify a particular month by adding it in yyyy-mm format after the requested metric (e.g. “viewsTotal/2019-02”).

### Retrieving Total Views for a Dataset

Please note that “viewsTotal” is a combination of “viewsTotalRegular” and “viewsTotalMachine” which can be requested separately.

```
curl "$DV_BASE_URL/api/datasets/:persistentId/makeDataCount/viewsTotal?persistentId=$DOI"
```

### Retrieving Unique Views for a Dataset

Please note that “viewsUnique” is a combination of “viewsUniqueRegular” and “viewsUniqueMachine” which can be requested separately.

```
curl "$DV_BASE_URL/api/datasets/:persistentId/makeDataCount/viewsUnique?persistentId=$DOI"
```

### Retrieving Total Downloads for a Dataset

Please note that “downloadsTotal” is a combination of “downloadsTotalRegular” and “downloadsTotalMachine” which can be requested separately.

```
curl "$DV_BASE_URL/api/datasets/:persistentId/makeDataCount/downloadsTotal?persistentId=$DOI"
```

### Retrieving Unique Downloads for a Dataset

Please note that “downloadsUnique” is a combination of “downloadsUniqueRegular” and “downloadsUniqueMachine” which can be requested separately.

```
curl "$DV_BASE_URL/api/datasets/:persistentId/makeDataCount/downloadsUnique?persistentId=$DOI"
```

### Retrieving Citations for a Dataset

```
curl "$DV_BASE_URL/api/datasets/:persistentId/makeDataCount/citations?persistentId=$DOI"
```

### Delete Unpublished Dataset

Delete the dataset whose id is passed:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X DELETE http://$SERVER/api/datasets/$id
```

## Delete Published Dataset

Normally published datasets should not be deleted, but there exists a “destroy” API endpoint for superusers which will act on a dataset given a persistent ID or dataset database ID:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X DELETE http://$SERVER/api/datasets/
:persistentId/destroy/?persistentId=doi:10.5072/FK2/AAA000
```

```
curl -H "X-Dataverse-key:$API_TOKEN" -X DELETE http://$SERVER/api/datasets/
999/destroy
```

Calling the destroy endpoint is permanent and irreversible. It will remove the dataset and its datafiles, then re-index the parent dataverse in Solr. This endpoint requires the API token of a superuser.

## 3.6.3 Files

### Adding Files

---

**Note:** Files can be added via the native API but the operation is performed on the parent object, which is a dataset. Please see the [Datasets](#) endpoint above for more information.

---

### Accessing (downloading) files

---

**Note:** Access API has its own section in the Guide: [Data Access API](#)

---

**Note** Data Access API calls can now be made using persistent identifiers (in addition to database ids). This is done by passing the constant `:persistentId` where the numeric id of the file is expected, and then passing the actual persistent id as a query parameter with the name `persistentId`.

Example: Getting the file whose DOI is *10.5072/FK2/J8SJZB*

```
GET http://$SERVER/api/access/datafile/:persistentId/?persistentId=doi:10.
↪5072/FK2/J8SJZB
```

### Restrict Files

Restrict or unrestrict an existing file where `id` is the database id of the file or `pid` is the persistent id (DOI or Handle) of the file to restrict. Note that some Dataverse installations do not allow the ability to restrict files.

A curl example using an `id`:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT -d true http://$SERVER/api/files/{id}/
↪restrict
```

A curl example using a `pid`:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X PUT -d true http://$SERVER/api/files/
↪:persistentId/restrict?persistentId={pid}
```



## Uningest a File

Reverse the tabular data ingest process performed on a file where `{id}` is the database id of the file to process. Note that this requires “superuser” credentials:

```
POST http://$SERVER/api/files/{id}/uningest?key={apiKey}
```

## Reingest a File

Attempt to ingest an existing datafile as tabular data. This API can be used on a file that was not ingested as tabular back when it was uploaded. For example, a Stata v.14 file that was uploaded before ingest support for Stata 14 was added (in Dataverse v.4.9). It can also be used on a file that failed to ingest due to a bug in the ingest plugin that has since been fixed (hence the name “reingest”).

Note that this requires “superuser” credentials:

```
POST http://$SERVER/api/files/{id}/reingest?key={apiKey}
```

(`{id}` is the database id of the file to process)

Note: at present, the API cannot be used on a file that’s already successfully ingested as tabular.

## Redetect File Type

Dataverse uses a variety of methods for determining file types (MIME types or content types) and these methods (listed below) are updated periodically. If you have files that have an unknown file type, you can have Dataverse attempt to redetect the file type.

When using the curl command below, you can pass `dryRun=true` if you don’t want any changes to be saved to the database. Change this to `dryRun=false` (or omit it) to save the change. In the example below, the file is identified by database id “42”.

```
export FILE_ID=42
curl -H "X-Dataverse-key:$API_TOKEN" -X POST $SERVER_URL/api/files/$FILE_ID/redetect?dryRun=true
```

Currently the following methods are used to detect file types:

- The file type detected by the browser (or sent via API).
- JHOVE: <http://jhove.openpreservation.org>
- As a last resort the file extension (e.g. “.ipybn”) is used, defined in a file called `MimeTypeDetectionByFileExtension.properties`.

## Replacing Files

Replace an existing file where `id` is the database id of the file to replace or `pid` is the persistent id (DOI or Handle) of the file. Requires the `file` to be passed as well as a `jsonString` expressing the new metadata. Note that metadata such as `description`, `directoryLabel` (File Path) and `tags` are not carried over from the file being replaced:

```
POST -F 'file=@file.extension' -F 'jsonData={json}' http://$SERVER/api/files/{id}/↵metadata?key={apiKey}
```

Example:

```
curl -H "X-Dataverse-key:$API_TOKEN" -X POST -F 'file=@data.tsv' \
-F 'jsonData={"description":"My description.", "categories":["Data"], "forceReplace
↪":false}' \
"https://demo.dataverse.org/api/files/$FILE_ID/replace"
```

### Getting File Metadata

Provides a json representation of the file metadata for an existing file where `id` is the database id of the file to replace or `pid` is the persistent id (DOI or Handle) of the file:

```
GET http://$SERVER/api/files/{id}/metadata
```

The current draft can also be viewed if you have permissions and pass your `apiKey`:

```
GET http://$SERVER/api/files/{id}/metadata/draft?key={apiKey}
```

Note: The `id` returned in the json response is the `id` of the file metadata version.

### Updating File Metadata

Updates the file metadata for an existing file where `id` is the database id of the file to replace or `pid` is the persistent id (DOI or Handle) of the file. Requires a `jsonString` expressing the new metadata. No metadata from the previous version of this file will be persisted, so if you want to update a specific field first get the json with the above command and alter the fields you want:

```
POST -F 'jsonData={json}' http://$SERVER/api/files/{id}/metadata?key={apiKey}
```

Example:

```
curl -H "X-Dataverse-key:{apiKey}" -X POST -F 'jsonData={"description":"My
↪description bbb.", "provFreeform":"Test prov freeform", "categories":["Data"],
↪"restrict":false}' 'http://localhost:8080/api/files/264/metadata'
```

Also note that `dataFileTags` are not versioned and changes to these will update the published version of the file.

### Editing Variable Level Metadata

Updates variable level metadata using `ddi xml $file`, where `$id` is file id:

```
PUT https://$SERVER/api/edit/$id --upload-file $file
```

Example: `curl -H "X-Dataverse-key:$API_TOKEN" -X PUT http://localhost:8080/api/edit/95 --upload-file dct.xml`

You can download `dct.xml` from the example above to see what the XML looks like.

### Provenance

Get Provenance JSON for an uploaded file:

```
GET http://$SERVER/api/files/{id}/prov-json?key=$apiKey
```

Get Provenance Description for an uploaded file:

```
GET http://$SERVER/api/files/{id}/prov-freeform?key=$apiKey
```

Create/Update Provenance JSON and provide related entity name for an uploaded file:

```
POST http://$SERVER/api/files/{id}/prov-json?key=$apiKey&entityName=$entity -H
↪ "Content-type:application/json" --upload-file $filePath
```

Create/Update Provenance Description for an uploaded file. Requires a JSON file with the description connected to a key named “text”:

```
POST http://$SERVER/api/files/{id}/prov-freeform?key=$apiKey -H "Content-
↪ type:application/json" --upload-file $filePath
```

Delete Provenance JSON for an uploaded file:

```
DELETE http://$SERVER/api/files/{id}/prov-json?key=$apiKey
```

## Datafile Integrity

Starting the release 4.10 the size of the saved original file (for an ingested tabular datafile) is stored in the database. The following API will retrieve and permanently store the sizes for any already existing saved originals:

```
GET http://$SERVER/api/admin/datafiles/integrity/fixmissingoriginalsizes(?limit=N)
```

Note the optional “limit” parameter. Without it, the API will attempt to populate the sizes for all the saved originals that don’t have them in the database yet. Otherwise it will do so for the first N such datafiles.

## 3.6.4 Builtin Users

Builtin users are known as “Username/Email and Password” users in the *Account Creation + Management* of the User Guide. Dataverse stores a password (encrypted, of course) for these users, which differs from “remote” users such as Shibboleth or OAuth users where the password is stored elsewhere. See also “Auth Modes: Local vs. Remote vs. Both” in the *Configuration* section of the Installation Guide. It’s a valid configuration of Dataverse to not use builtin users at all.

### Create a Builtin User

For security reasons, builtin users cannot be created via API unless the team who runs the Dataverse installation has populated a database setting called `BuiltinUsers.KEY`, which is described under “Securing Your Installation” and “Database Settings” in the *Configuration* section of the Installation Guide. You will need to know the value of `BuiltinUsers.KEY` before you can proceed.

To create a builtin user via API, you must first construct a JSON document. You can download `user-add.json` or copy the text below as a starting point and edit as necessary.

```
{
  "firstName": "Lisa",
  "lastName": "Simpson",
  "userName": "lsimpson",
  "affiliation": "Springfield",
  "position": "Student",
```

```
"email": "lsimpson@mailinator.com"
}
```

Place this `user-add.json` file in your current directory and run the following `curl` command, substituting variables as necessary. Note that both the password of the new user and the value of `BuiltinUsers.KEY` are passed as query parameters:

```
curl -d @user-add.json -H "Content-type:application/json" "$SERVER_URL/api/builtin-
↪users?password=$NEWUSER_PASSWORD&key=$BUILTIN_USERS_KEY"
```

### 3.6.5 Roles

#### Create a New Role in a Dataverse

Creates a new role in dataverse object whose `Id` is `dataverseIdtf` (that's an `id/alias`):

```
POST http://$SERVER/api/roles?dvo=$dataverseIdtf&key=$apiKey
```

#### Show Role

Shows the role with `id`:

```
GET http://$SERVER/api/roles/$id
```

#### Delete Role

Deletes the role with `id`:

```
DELETE http://$SERVER/api/roles/$id
```

### 3.6.6 Explicit Groups

#### Create New Explicit Group

Explicit groups list their members explicitly. These groups are defined in dataverses, which is why their API endpoint is under `api/dataverses/$id/`, where `$id` is the `id` of the dataverse.

Create a new explicit group under dataverse `$id`:

```
POST http://$server/api/dataverses/$id/groups
```

Data being POSTed is json-formatted description of the group:

```
{
  "description": "Describe the group here",
  "displayName": "Close Collaborators",
  "aliasInOwner": "ccs"
}
```

### List Explicit Groups in a Dataverse

List explicit groups under dataverse \$id:

```
GET http://$server/api/dataverses/$id/groups
```

### Show Single Group in a Dataverse

Show group \$groupAlias under dataverse \$dv:

```
GET http://$server/api/dataverses/$dv/groups/$groupAlias
```

### Update Group in a Dataverse

Update group \$groupAlias under dataverse \$dv. The request body is the same as the create group one, except that the group alias cannot be changed. Thus, the field `aliasInOwner` is ignored.

```
PUT http://$server/api/dataverses/$dv/groups/$groupAlias
```

### Delete Group from a Dataverse

Delete group \$groupAlias under dataverse \$dv:

```
DELETE http://$server/api/dataverses/$dv/groups/$groupAlias
```

### Add Multiple Role Assignees to an Explicit Group

Bulk add role assignees to an explicit group. The request body is a JSON array of role assignee identifiers, such as `@admin`, `&ip/localhosts` or `:authenticated-users`:

```
POST http://$server/api/dataverses/$dv/groups/$groupAlias/roleAssignees
```

### Add a Role Assignee to an Explicit Group

Add a single role assignee to a group. Request body is ignored:

```
PUT http://$server/api/dataverses/$dv/groups/$groupAlias/roleAssignees/  
↪$roleAssigneeIdentifier
```

### Remove a Role Assignee from an Explicit Group

Remove a single role assignee from an explicit group:

```
DELETE http://$server/api/dataverses/$dv/groups/$groupAlias/roleAssignees/  
↪$roleAssigneeIdentifier
```

### 3.6.7 Shibboleth Groups

Management of Shibboleth groups via API is documented in the *Shibboleth* section of the Installation Guide.

### 3.6.8 Info

#### Show Dataverse Version and Build Number

Get the Dataverse version. The response contains the version and build numbers:

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of export below.

---

```
export SERVER_URL=https://demo.dataverse.org
curl $SERVER_URL/api/info/version
```

The fully expanded example above (without environment variables) looks like this:

```
curl https://demo.dataverse.org/api/info/version
```

#### Show Dataverse Server Name

Get the server name. This is useful when a Dataverse system is composed of multiple Java EE servers behind a load balancer:

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of export below.

---

```
export SERVER_URL=https://demo.dataverse.org
curl $SERVER_URL/api/info/server
```

The fully expanded example above (without environment variables) looks like this:

```
curl https://demo.dataverse.org/api/info/server
```

#### Show Custom Popup Text for Publishing Datasets

For now, only the value for the `:DatasetPublishPopupCustomText` setting from the *Configuration* section of the Installation Guide is exposed:

---

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of export below.

---

```
export SERVER_URL=https://demo.dataverse.org
curl $SERVER_URL/api/info/settings/:DatasetPublishPopupCustomText
```

The fully expanded example above (without environment variables) looks like this:

```
curl https://demo.dataverse.org/api/info/settings/:DatasetPublishPopupCustomText
```

### Get API Terms of Use URL

Get API Terms of Use. The response contains the text value inserted as API Terms of use which uses the database setting :ApiTermsOfUse:

**Note:** See *curl Examples and Environment Variables* if you are unfamiliar with the use of export below.

```
export SERVER_URL=https://demo.dataverse.org
curl $SERVER_URL/api/info/apiTermsOfUse
```

The fully expanded example above (without environment variables) looks like this:

```
curl https://demo.dataverse.org/api/info/apiTermsOfUse
```

## 3.6.9 Metadata Blocks

### Show Info About All Metadata Blocks

Lists brief info about all metadata blocks registered in the system:

```
GET http://$SERVER/api/metadatablocks
```

### Show Info About Single Metadata Block

Return data about the block whose `identifier` is passed. `identifier` can either be the block's id, or its name:

```
GET http://$SERVER/api/metadatablocks/$identifier
```

## 3.6.10 Notifications

### Get All Notifications by User

Each user can get a dump of their notifications by passing in their API token:

```
curl -H "X-Dataverse-key:$API_TOKEN" $SERVER_URL/api/notifications/all
```

## 3.6.11 Admin

This is the administrative part of the API. For security reasons, it is absolutely essential that you block it before allowing public access to a Dataverse installation. Blocking can be done using settings. See the `post-install-api-block.sh` script in the `scripts/api` folder for details. See also “Blocking API Endpoints” under “Securing Your Installation” in the *Configuration* section of the Installation Guide.

### List All Database Settings

List all settings:

```
GET http://$SERVER/api/admin/settings
```

### Configure Database Setting

Sets setting name to the body of the request:

```
PUT http://$SERVER/api/admin/settings/$name
```

### Get Single Database Setting

Get the setting under name:

```
GET http://$SERVER/api/admin/settings/$name
```

### Delete Database Setting

Delete the setting under name:

```
DELETE http://$SERVER/api/admin/settings/$name
```

### List Authentication Provider Factories

List the authentication provider factories. The alias field of these is used while configuring the providers themselves.

```
GET http://$SERVER/api/admin/authenticationProviderFactories
```

### List Authentication Providers

List all the authentication providers in the system (both enabled and disabled):

```
GET http://$SERVER/api/admin/authenticationProviders
```

### Add Authentication Provider

Add new authentication provider. The POST data is in JSON format, similar to the JSON retrieved from this command's GET counterpart.

```
POST http://$SERVER/api/admin/authenticationProviders
```



### Show Authentication Provider

Show data about an authentication provider:

```
GET http://$SERVER/api/admin/authenticationProviders/$id
```

### Enable or Disable an Authentication Provider

Enable or disable an authentication provider (denoted by id):

```
PUT http://$SERVER/api/admin/authenticationProviders/$id/enabled
```

**Note:** The former endpoint, ending with `:enabled` (that is, with a colon), is still supported, but deprecated.

### Check If an Authentication Provider is Enabled

Check whether an authentication provider is enabled:

```
GET http://$SERVER/api/admin/authenticationProviders/$id/enabled
```

The body of the request should be either `true` or `false`. Content type has to be `application/json`, like so:

```
curl -H "Content-type: application/json" -X POST -d"false" http://localhost:8080/api/  
↪admin/authenticationProviders/echo-dignified/:enabled
```

### Delete an Authentication Provider

Deletes an authentication provider from the system. The command succeeds even if there is no such provider, as the postcondition holds: there is no provider by that id after the command returns.

```
DELETE http://$SERVER/api/admin/authenticationProviders/$id/
```

### List Global Roles

List all global roles in the system.

```
GET http://$SERVER/api/admin/roles
```

### Create Global Role

Creates a global role in the Dataverse installation. The data POSTed are assumed to be a role JSON.

```
POST http://$SERVER/api/admin/roles
```

## List Users

List users with the options to search and “page” through results. Only accessible to superusers. Optional parameters:

- `searchTerm` A string that matches the beginning of a user identifier, first name, last name or email address.
- `itemsPerPage` The number of detailed results to return. The default is 25. This number has no limit. e.g. You could set it to 1000 to return 1,000 results
- `selectedPage` The page of results to return. The default is 1.

```
GET http://$SERVER/api/admin/list-users
```

Sample output appears below.

- When multiple pages of results exist, the `selectedPage` parameters may be specified.
- Note, the resulting pagination section includes `pageCount`, `previousPageNumber`, `nextPageNumber`, and other variables that may be used to re-create the UI.

```
{
  "status": "OK",
  "data": {
    "userCount": 27,
    "selectedPage": 1,
    "pagination": {
      "isNecessary": true,
      "numResults": 27,
      "numResultsString": "27",
      "docsPerPage": 25,
      "selectedPageNumber": 1,
      "pageCount": 2,
      "hasPreviousPageNumber": false,
      "previousPageNumber": 1,
      "hasNextPageNumber": true,
      "nextPageNumber": 2,
      "startResultNumber": 1,
      "endResultNumber": 25,
      "startResultNumberString": "1",
      "endResultNumberString": "25",
      "remainingResults": 2,
      "numberNextResults": 2,
      "pageNumberList": [
        1,
        2
      ]
    },
    "bundleStrings": {
      "userId": "ID",
      "userIdentifier": "Username",
      "lastName": "Last Name ",
      "firstName": "First Name ",
      "email": "Email",
      "affiliation": "Affiliation",
      "position": "Position",
      "isSuperuser": "Superuser",
      "authenticationProvider": "Authentication",
      "roles": "Roles",
      "createdTime": "Created Time",
      "lastLoginTime": "Last Login Time",
    }
  }
}
```

```

        "lastApiUseTime":"Last API Use Time"
    },
    "users":[
        {
            "id":8,
            "userIdentifier":"created1",
            "lastName":"created1",
            "firstName":"created1",
            "email":"created1@g.com",
            "affiliation":"hello",
            "isSuperuser":false,
            "authenticationProvider":"BuiltinAuthenticationProvider",
            "roles":"Curator",
            "createdTime":"2017-06-28 10:36:29.444"
        },
        {
            "id":9,
            "userIdentifier":"created8",
            "lastName":"created8",
            "firstName":"created8",
            "email":"created8@g.com",
            "isSuperuser":false,
            "authenticationProvider":"BuiltinAuthenticationProvider",
            "roles":"Curator",
            "createdTime":"2000-01-01 00:00:00.0"
        },
        {
            "id":1,
            "userIdentifier":"dataverseAdmin",
            "lastName":"Admin",
            "firstName":"Dataverse",
            "email":"dataverse@mailinator2.com",
            "affiliation":"Dataverse.org",
            "position":"Admin",
            "isSuperuser":true,
            "authenticationProvider":"BuiltinAuthenticationProvider",
            "roles":"Admin, Contributor",
            "createdTime":"2000-01-01 00:00:00.0",
            "lastLoginTime":"2017-07-03 12:22:35.926",
            "lastApiUseTime":"2017-07-03 12:55:57.186"
        }
        // ... 22 more user documents ...
    ]
}

```

**Note:** “List all users” GET [http://\\$SERVER/api/admin/authenticatedUsers](http://$SERVER/api/admin/authenticatedUsers) is deprecated, but supported.

### List Single User

List user whose identifier (without the @ sign) is passed:

```
GET http://$SERVER/api/admin/authenticatedUsers/$identifier
```

Sample output using “dataverseAdmin” as the identifier:

```
{
  "authenticationProviderId": "builtin",
  "persistentUserId": "dataverseAdmin",
  "position": "Admin",
  "id": 1,
  "identifier": "@dataverseAdmin",
  "displayName": "Dataverse Admin",
  "firstName": "Dataverse",
  "lastName": "Admin",
  "email": "dataverse@mailinator.com",
  "superuser": true,
  "affiliation": "Dataverse.org"
}
```

Create an authenticateUser:

```
POST http://$SERVER/api/admin/authenticatedUsers
```

POSTed JSON example:

```
{
  "authenticationProviderId": "orcid",
  "persistentUserId": "0000-0002-3283-0661",
  "identifier": "@pete",
  "firstName": "Pete K.",
  "lastName": "Dataversky",
  "email": "pete@mailinator.com"
}
```

### Merge User Accounts

If a user has created multiple accounts and has been performed actions under both accounts that need to be preserved, these accounts can be combined. One account can be merged into another account and all data associated with both accounts will be combined in the surviving account. Only accessible to superusers.:

```
POST https://$SERVER/api/users/$toMergeIdentifier/mergeIntoUser/$continuingIdentifier
```

Example: `curl -H "X-Dataverse-key: $API_TOKEN" -X POST http://demo.dataverse.org/api/users/jsmith2/mergeIntoUser/jsmith`

This action moves account data from jsmith2 into the account jsmith and deletes the account of jsmith2.

### Change User Identifier

Changes identifier for user in AuthenticatedUser, BuiltinUser, AuthenticatedUserLookup & RoleAssignment. Allows them to log in with the new identifier. Only accessible to superusers.:

```
PUT http://$SERVER/api/users/$oldIdentifier/changeIdentifier/$newIdentifier
```

Example: `curl -H "X-Dataverse-key: $API_TOKEN" -X POST https://demo.dataverse.org/api/users/johnsmith/changeIdentifier/jsmith`

This action changes the identifier of user johnsmith to jsmith.

### Make User a SuperUser

Toggles superuser mode on the `AuthenticatedUser` whose identifier (without the @ sign) is passed.

```
POST http://$SERVER/api/admin/superuser/$identifier
```

### List Role Assignments of a Role Assignee

List all role assignments of a role assignee (i.e. a user or a group):

```
GET http://$SERVER/api/admin/assignments/assignees/$identifier
```

Note that `identifier` can contain slashes (e.g. `&ip/localhost-users`).

### List Permissions a User Has on a Dataverse or Dataset

List permissions a user (based on API Token used) has on a dataverse or dataset:

```
GET http://$SERVER/api/admin/permissions/$identifier
```

The `$identifier` can be a dataverse alias or database id or a dataset persistent ID or database id.

### Show Role Assignee

List a role assignee (i.e. a user or a group):

```
GET http://$SERVER/api/admin/assignee/$identifier
```

The `$identifier` should start with an @ if it's a user. Groups start with &. "Built in" users and groups start with :. Private URL users start with #.

### Saved Search

The Saved Search, Linked Dataverses, and Linked Datasets features shipped with Dataverse 4.0, but as a "superuser-only" because they are **experimental** (see #1364, #1813, #1840, #1890, #1939, #2167, #2186, #2053, and #2543). The following API endpoints were added to help people with access to the "admin" API make use of these features in their current form. Of particular interest should be the "makelinks" endpoint because it needs to be called periodically (via cron or similar) to find new dataverses and datasets that match the saved search and then link the search results to the dataverse in which the saved search is defined (#2531 shows an example). There is a known issue (#1364) that once a link to a dataverse or dataset is created, it cannot be removed (apart from database manipulation and reindexing) which is why a DELETE endpoint for saved searches is neither documented nor functional. The Linked Dataverses feature is **powered by Saved Search** and therefore requires that the "makelinks" endpoint be executed on a periodic basis as well.

List all saved searches.

```
GET http://$SERVER/api/admin/savedsearches/list
```

List a saved search by database id.

```
GET http://$SERVER/api/admin/savedsearches/$id
```

Execute a saved search by database id and make links to dataverses and datasets that are found. The JSON response indicates which dataverses and datasets were newly linked versus already linked. The `debug=true` query parameter adds to the JSON response extra information about the saved search being executed (which you could also get by listing the saved search).

```
PUT http://$SERVER/api/admin/savedsearches/makelinks/$id?debug=true
```

Execute all saved searches and make links to dataverses and datasets that are found. `debug` works as described above.

```
PUT http://$SERVER/api/admin/savedsearches/makelinks/all?debug=true
```

### Dataset Integrity

Recalculate the UNF value of a dataset version, if it's missing, by supplying the dataset version database id:

```
POST http://$SERVER/api/admin/datasets/integrity/{datasetVersionId}/fixmissingunf
```

### Datafile Integrity

Recalculate the check sum value value of a datafile, by supplying the file's database id and an algorithm (Valid values for `$ALGORITHM` include MD5, SHA-1, SHA-256, and SHA-512):

```
curl -H X-Dataverse-key:$API_TOKEN -X POST $SERVER_URL/api/admin/  
↪computeDataFileHashValue/{fileId}/algorithm/$ALGORITHM
```

Validate an existing check sum value against one newly calculated from the saved file:

```
curl -H X-Dataverse-key:$API_TOKEN -X POST $SERVER_URL/api/admin/  
↪validateDataFileHashValue/{fileId}
```

These are only available to super users.

### Dataset Validation

Validate the dataset and its components (DatasetVersion, FileMetadatas, etc.) for constraint violations:

```
curl $SERVER_URL/api/admin/validate/dataset/{datasetId}
```

if validation fails, will report the specific database entity and the offending value. For example:

```
{"status":"OK","data":{"entityClassDatabaseTableRowId":["DatasetVersion id:73"],"field  
↪":"archiveNote","invalidValue":"random text, not a url"}}
```

If the optional argument `variables=true` is specified, the API will also validate the metadata associated with any tabular data files found in the dataset specified. (For example: an invalid or empty variable name).

Validate all the datasets in the Dataverse, report any constraint violations found:

```
curl $SERVER_URL/api/admin/validate/datasets
```

If the optional argument `variables=true` is specified, the API will also validate the metadata associated with any tabular data files. (For example: an invalid or empty variable name). Note that validating all the tabular metadata may significantly increase the run time of the full validation pass.

This API streams its output in real time, i.e. it will start producing the output immediately and will be reporting on the progress as it validates one dataset at a time. For example:

```
{ "datasets": [
  { "datasetId": 27, "status": "valid" },
  { "datasetId": 29, "status": "valid" },
  { "datasetId": 31, "status": "valid" },
  { "datasetId": 33, "status": "valid" },
  { "datasetId": 35, "status": "valid" },
  { "datasetId": 41, "status": "invalid", "entityClassDatabaseTableRowId":
↵ "[DatasetVersion id:73]", "field": "archiveNote", "invalidValue": "random text, not a
↵ url" },
  { "datasetId": 57, "status": "valid" }
]
```

Note that if you are attempting to validate a very large number of datasets in your Dataverse, this API may time out - subject to the timeout limit set in your Glassfish configuration. If this is a production Dataverse instance serving large amounts of data, you most likely have that timeout set to some high value already. But if you need to increase it, it can be done with the `asadmin` command. For example:

```
asadmin set server-config.network-config.protocols.protocol.http-listener-1.http
↵ request-timeout-seconds=3600
```

## Workflows

List all available workflows in the system:

```
GET http://$SERVER/api/admin/workflows
```

Get details of a workflow with a given id:

```
GET http://$SERVER/api/admin/workflows/$id
```

Add a new workflow. Request body specifies the workflow properties and steps in JSON format. Sample json files are available at `scripts/api/data/workflows/`:

```
POST http://$SERVER/api/admin/workflows
```

Delete a workflow with a specific id:

```
DELETE http://$SERVER/api/admin/workflows/$id
```

**Warning:** If the workflow designated by `$id` is a default workflow, a 403 FORBIDDEN response will be returned, and the deletion will be canceled.

List the default workflow for each trigger type:

```
GET http://$SERVER/api/admin/workflows/default/
```

Set the default workflow for a given trigger. This workflow is run when a dataset is published. The body of the PUT request is the id of the workflow. Trigger types are `PrePublishDataset`, `PostPublishDataset`:

```
PUT http://$SERVER/api/admin/workflows/default/$triggerType
```

Get the default workflow for `triggerType`. Returns a JSON representation of the workflow, if present, or 404 NOT FOUND.

```
GET http://$SERVER/api/admin/workflows/default/$triggerType
```

Unset the default workflow for `triggerType`. After this call, dataset releases are done with no workflow.

```
DELETE http://$SERVER/api/admin/workflows/default/$triggerType
```

Set the whitelist of IP addresses separated by a semicolon (;) allowed to resume workflows. Request body is a list of IP addresses allowed to send “resume workflow” messages to this Dataverse instance:

```
PUT http://$SERVER/api/admin/workflows/ip-whitelist
```

Get the whitelist of IP addresses allowed to resume workflows:

```
GET http://$SERVER/api/admin/workflows/ip-whitelist
```

Restore the whitelist of IP addresses allowed to resume workflows to default (localhost only):

```
DELETE http://$SERVER/api/admin/workflows/ip-whitelist
```

## Metrics

Clear all cached metric results:

```
DELETE http://$SERVER/api/admin/clearMetricsCache
```

Clear a specific metric cache. Currently this must match the name of the row in the table, which is named `metricName*_*metricYYYYMM` (or just `metricName` if there is no date range for the metric). For example `dataversesToMonth_2018-05`:

```
DELETE http://$SERVER/api/admin/clearMetricsCache/$metricDbName
```

## Inherit Dataverse Role Assignments

`Recursively` applies the role assignments of the specified dataverse, for the roles specified by the `:InheritParentRoleAssignments` setting, to all dataverses contained within it:

```
GET http://$SERVER/api/admin/dataverse/{dataverse alias}/addRoleAssignmentsToChildren
```

**Note:** setting `:InheritParentRoleAssignments` will automatically trigger inheritance of the parent dataverse’s role assignments for a newly created dataverse. Hence this API call is intended as a way to update existing child dataverses or to update children after a change in role assignments has been made on a parent dataverse.



## 3.7 Metrics API

The Metrics API provides counts of downloads, datasets created, files uploaded, and more, as described below. Dataverse also supports Make Data Count, which is described in the *Make Data Count* section of the Admin Guide.

### Contents:

- *Total*
- *To-Month*
- *Past Days*
- *Dataverse Specific Metrics*
  - *By Subject*
  - *By Category*
- *Dataset Specific Metrics*
  - *By Subject*
  - *By Subject, and to Month*
- *Metric Query Parameters*
  - *dataLocation*

**Note:** The Metrics API can be used from scripts running in web browsers, as it allows cross-origin resource sharing (CORS).

**Note:** For all metrics *besides* Past Days Count (`/pastDays/$days`), Database setting `MetricsCacheTimeoutMinutes` defines how long the cached value will be returned by subsequent queries.

### 3.7.1 Total

Returns a count of various objects in dataverse over all-time:

```
GET https://$SERVER/api/info/metrics/$type
```

`$type` can be set to `dataverses`, `datasets`, `files` or `downloads`.

Example: `curl https://demo.dataverse.org/api/info/metrics/downloads`

### 3.7.2 To-Month

Returns a count of various objects in dataverse up to a specified month `YYYY-DD` in `YYYY-MM` format (e.g. `2018-01`):

```
GET https://$SERVER/api/info/metrics/$type/toMonth/$YYYY-DD
```

\$type can be set to dataverses, datasets, files or downloads.

Example: `curl https://demo.dataverse.org/api/info/metrics/dataverses/toMonth/2018-01`

### 3.7.3 Past Days

Returns a count of various objects in dataverse for the past \$days (e.g. 30):

```
GET https://$SERVER/api/info/metrics/$type/pastDays/$days
```

\$type can be set to dataverses, datasets, files or downloads.

Example: `curl https://demo.dataverse.org/api/info/metrics/datasets/pastDays/30`

### 3.7.4 Dataverse Specific Metrics

#### By Subject

Returns the number of dataverses by each subject:

```
GET https://$SERVER/api/info/metrics/dataverses/bySubject
```

#### By Category

Returns the number of dataverses by each category:

```
GET https://$SERVER/api/info/metrics/dataverses/byCategory
```

### 3.7.5 Dataset Specific Metrics

#### By Subject

Returns the number of datasets by each subject:

```
GET https://$SERVER/api/info/metrics/datasets/bySubject
```

#### By Subject, and to Month

Returns the number of datasets by each subject, and up to a specified month \$YYYY-DD in YYYY-MM format (e.g. 2018-01):

```
GET https://$SERVER/api/info/metrics/datasets/bySubject/toMonth/$YYYY-DD
```

Example: `curl https://demo.dataverse.org/api/info/metrics/datasets/bySubject/toMonth/2018-01`

### 3.7.6 Metric Query Parameters

To further tailor your metric, query parameters can be provided.

## dataLocation

Specifies whether the metric should query local data, remote data (e.g. harvested), or all data when getting results. Only works for dataset metrics.

Example: `curl https://demo.dataverse.org/api/info/metrics/datasets/?dataLocation=remote`

## 3.8 SWORD API

**SWORD** stands for “Simple Web-service Offering Repository Deposit” and is a “profile” of AtomPub ([RFC 5023](#)) which is a RESTful API that allows non-Dataverse software to deposit files and metadata into a Dataverse installation. *Client libraries* are available in Python, Javascript, Java, R, Ruby, and PHP.

### Contents:

- *About*
- *Backward incompatible changes*
- *New features as of v1.1*
- *curl examples*
  - *Retrieve SWORD service document*
  - *Create a dataset with an Atom entry*
    - \* *Dublin Core Terms (DC Terms) Qualified Mapping - Dataverse DB Element Crosswalk*
  - *List datasets in a dataverse*
  - *Add files to a dataset with a zip file*
  - *Display a dataset atom entry*
  - *Display a dataset statement*
  - *Delete a file by database id*
  - *Replacing metadata for a dataset*
  - *Delete a dataset*
  - *Determine if a dataverse has been published*
  - *Publish a dataverse*
  - *Publish a dataset*
- *Known issues*
- *Bug fixes in v1.1*
- *Client libraries*

### 3.8.1 About

Introduced in Dataverse Network (DVN) 3.6, the SWORD API was formerly known as the “Data Deposit API” and `data-deposit/v1` appeared in the URLs. For backwards compatibility these URLs continue to work (with

deprecation warnings). Due to architectural changes and security improvements (especially the introduction of API tokens) in Dataverse 4.0, a few backward incompatible changes were necessarily introduced and for this reason the version has been increased to `v1.1`. For details, see *Backward incompatible changes*.

Dataverse implements most of `SWORDv2`, which is specified at <http://swordapp.github.io/SWORDv2-Profile/SWORDProfile.html>. Please reference the `SWORDv2` specification for expected HTTP status codes (i.e. 201, 204, 404, etc.), headers (i.e. “Location”), etc.

As a profile of AtomPub, XML is used throughout SWORD. As of Dataverse 4.0 datasets can also be created via JSON using the “native” API. SWORD is limited to the dozen or so fields listed below in the crosswalk, but the native API allows you to populate all metadata fields available in Dataverse.

### 3.8.2 Backward incompatible changes

For better security than in DVN 3.x, usernames and passwords are no longer accepted. The use of an API token is required.

Differences in Dataverse 4 from DVN 3.x lead to a few minor backward incompatible changes in the Dataverse implementation of SWORD, which are listed below. Old `v1` URLs should continue to work but the `Service Document` will contain a deprecation warning and responses will contain `v1.1` URLs. See also *Known issues*.

- Newly required fields when creating/editing datasets for compliance with the [Joint Declaration for Data Citation principles](#).
  - `dcterms:creator` (maps to `authorName`)
  - `dcterms:description`
- Deaccessioning is no longer supported. An alternative will be developed at <https://github.com/IQSS/dataverse/issues/778>
- The Service Document will show a single API Terms of Use rather than root level and dataverse level Deposit Terms of Use.

### 3.8.3 New features as of v1.1

- Dataverse 4 supports API tokens and requires them to be used for APIs instead of a username and password. In the `curl` examples below, you will see `curl -u $API_TOKEN:` showing that you should send your API token as the username and nothing as the password. For example, `curl -u 54b143b5-d001-4254-afc0-a1c0f6a5b5a7:.`
- SWORD operations no longer require “admin” permission. In order to use any SWORD operation in DVN 3.x, you had to be an “admin” on a dataverse (the container for your dataset) and similar rules were applied in Dataverse 4.4 and earlier (the `EditDataverse` permission was required). The SWORD API has now been fully integrated with the Dataverse 4 permission model such that any action you have permission to perform in the GUI or “native” API you are able to perform via SWORD. This means that even a user with a “Contributor” role can operate on datasets via SWORD. Note that users with the “Contributor” role do not have the `PublishDataset` permission and will not be able publish their datasets via any mechanism, GUI or API.
- Dataverses can be published via SWORD.
- Datasets versions will only be increased to the next minor version (i.e. 1.1) rather than a major version (2.0) if possible. This depends on the nature of the change. Adding or removing a file, for example, requires a major version bump.
- “Author Affiliation” can now be populated with an XML attribute. For example: `<dcterms:creator affiliation="Coffee Bean State University">Stumptown, Jane</dcterms:creator>`

- “Contributor” can now be populated and the “Type” (Editor, Funder, Researcher, etc.) can be specified with an XML attribute. For example: `<dcterms:contributor type="Funder">CaffeineForAll</dcterms:contributor>`
- “License” can now be set with `dcterms:license` and the possible values are “CC0” and “NONE”. “License” interacts with “Terms of Use” (`dcterms:rights`) in that if you include `dcterms:rights` in the XML, the license will be set to “NONE”. If you don’t include `dcterms:rights`, the license will default to “CC0”. It is invalid to specify “CC0” as a license and also include `dcterms:rights`; an error will be returned. For backwards compatibility, `dcterms:rights` is allowed to be blank (i.e. `<dcterms:rights></dcterms:rights>`) but blank values will not be persisted to the database and the license will be set to “NONE”.
- “Contact E-mail” is automatically populated from dataset owner’s email.
- “Subject” uses our controlled vocabulary list of subjects. This list is in the Citation Metadata of our User Guide > [Metadata References](#). Otherwise, if a term does not match our controlled vocabulary list, it will put any subject terms in “Keyword”. If Subject is empty it is automatically populated with “N/A”.
- Zero-length files are now allowed (but not necessarily encouraged).
- “Depositor” and “Deposit Date” are auto-populated.

### 3.8.4 curl examples

#### Retrieve SWORD service document

The service document enumerates the dataverses (“collections” from a SWORD perspective) the user can deposit data into. The “collectionPolicy” element for each dataverse contains the Terms of Use. Any user with an API token can use this API endpoint. Institution-wide Shibboleth groups are not respected because membership in such a group can only be set via a browser.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/
service-document
```

#### Create a dataset with an Atom entry

To create a dataset, you must have the “Dataset Creator” role (the `AddDataset` permission) on a dataverse. Practically speaking, you should first retrieve the service document to list the dataverses into which you are authorized to deposit data.

```
curl -u $API_TOKEN: --data-binary "@path/to/atom-entry-study.xml" -H
"Content-Type: application/atom+xml" https://$HOSTNAME/dvn/api/data-deposit/
v1.1/swordv2/collection/dataverse/$DATAVERSE_ALIAS
```

#### Example Atom entry (XML)

```
<?xml version="1.0"?>
<entry xmlns="http://www.w3.org/2005/Atom"
  xmlns:dcterms="http://purl.org/dc/terms/">
  <!-- some embedded metadata -->
  <dcterms:title>Roasting at Home</dcterms:title>
  <dcterms:creator>Peets, John</dcterms:creator>
  <dcterms:creator affiliation="Coffee Bean State University">Stumptown, Jane</
  ↪dcterms:creator>
  <!-- Dataverse controlled vocabulary subject term -->
  <dcterms:subject>Chemistry</dcterms:subject>
  <!-- keywords -->
  <dcterms:subject>coffee</dcterms:subject>
  <dcterms:subject>beverage</dcterms:subject>
```

```
<dcterms:subject>caffeine</dcterms:subject>
<dcterms:description>Considerations before you start roasting your own coffee at_
↪home.</dcterms:description>
<!-- Producer with financial or admin responsibility of the data -->
<dcterms:publisher>Coffee Bean State University</dcterms:publisher>
<dcterms:contributor type="Funder">CaffeineForAll</dcterms:contributor>
<!-- production date -->
<dcterms:date>2013-07-11</dcterms:date>
<!-- kind of data -->
<dcterms:type>aggregate data</dcterms:type>
<!-- List of sources of the data collection-->
<dcterms:source>Stumptown, Jane. 2011. Home Roasting. Coffeemill Press.</
↪dcterms:source>
<!-- related materials -->
<dcterms:relation>Peets, John. 2010. Roasting Coffee at the Coffee Shop._
↪Coffeemill Press</dcterms:relation>
<!-- geographic coverage -->
<dcterms:coverage>United States</dcterms:coverage>
<dcterms:coverage>Canada</dcterms:coverage>
<!-- license and restrictions -->
<dcterms:license>NONE</dcterms:license>
<dcterms:rights>Downloader will not use the Materials in any way prohibited by_
↪applicable laws.</dcterms:rights>
<!-- related publications -->
<dcterms:isReferencedBy holdingsURI="http://dx.doi.org/10.1038/dvn333" agency="DOI
↪" IDNo="10.1038/dvn333">Peets, J., & Stumptown, J. (2013). Roasting at Home._
↪New England Journal of Coffee, 3(1), 22-34.</dcterms:isReferencedBy>
</entry>
```

## Dublin Core Terms (DC Terms) Qualified Mapping - Dataverse DB Element Crosswalk

DC (terms: namespace)	Dataverse DB Element	Required	Note
dc-terms:title	title	Y	Title of the Dataset.
dc-terms:creator	authorName (LastName, FirstName)	Y	Author(s) for the Dataset.
dc-terms:subject	subject (Controlled Vocabulary) OR keyword	Y	Controlled Vocabulary list is in our User Guide > <a href="#">Metadata References</a> .
dc-terms:description	dsDescriptionValue	Y	Describing the purpose, scope or nature of the Dataset. Can also use dcterms:abstract.
dc-terms:publisher	producerName		Person or agency financially or administratively responsible for the Dataset
dc-terms:contributor	datasetContactEmail	Y	Contact Email is required so will need to add an attribute type="Contact". Also used for Funder: add attribute type="Funder" which maps to contributorName.
dc-terms:date	productionDate (YYYY-MM-DD or YYYY-MM or YYYY)		Production date of Dataset.
dc-terms:type	kindOfData		Type of data included in the file: survey data, census/enumeration data, aggregate data, clinical.
dc-terms:source	dataSources		List of books, articles, data files if any that served as the sources for the Dataset.
dc-terms:relation	relatedMaterial		Any related material (journal article citation is not included here - see: dcterms:isReferencedBy below).
dc-terms:coverage	otherGeographicCoverage		General information on the geographic coverage of the Dataset.
dc-terms:license	license		Set the license to CC0 (default in Dataverse for new Datasets), otherwise enter "NONE" and fill in the dcterms:rights field.
dc-terms:rights	termsOfUse		If not using CC0, enter any terms of use or restrictions for the Dataset.
dc-terms:isReferencedBy	publicationCitation		The publication (journal article, book, other work) that uses this dataset (include citation, permanent identifier (DOI), and permanent URL).

## List datasets in a dataverse

You must have permission to add datasets in a dataverse (the dataverse should appear in the service document) to list the datasets inside. Institution-wide Shibboleth groups are not respected because membership in such a group can only be set via a browser.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/collection/dataverse/$DATAVERSE_ALIAS
```

## Add files to a dataset with a zip file

You must have `EditDataset` permission (Contributor role or above such as Curator or Admin) on the dataset to add files.

```
curl -u $API_TOKEN: --data-binary @path/to/example.zip -H
"Content-Disposition: filename=example.zip" -H "Content-Type: application/
zip" -H "Packaging: http://purl.org/net/sword/package/SimpleZip" https://
$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit-media/study/doi:TEST/12345
```

### Display a dataset atom entry

You must have `ViewUnpublishedDataset` permission (Contributor role or above such as Curator or Admin) on the dataset to view its Atom entry.

Contains data citation (`bibliographicCitation`), alternate URI (persistent URI of study), edit URI, edit media URI, statement URI.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit/
study/doi:TEST/12345
```

### Display a dataset statement

Contains title, author, feed of file entries, `latestVersionState`, `locked` boolean, updated timestamp. You must have `ViewUnpublishedDataset` permission (Contributor role or above such as Curator or Admin) on the dataset to display the statement.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/
statement/study/doi:TEST/12345
```

### Delete a file by database id

You must have `EditDataset` permission (Contributor role or above such as Curator or Admin) on the dataset to delete files.

```
curl -u $API_TOKEN: -X DELETE https://$HOSTNAME/dvn/api/data-deposit/v1.1/
swordv2/edit-media/file/123
```

### Replacing metadata for a dataset

Please note that **ALL** metadata (title, author, etc.) will be replaced, including fields that can not be expressed with “dcterms” fields. You must have `EditDataset` permission (Contributor role or above such as Curator or Admin) on the dataset to replace metadata.

```
curl -u $API_TOKEN: --upload-file "path/to/atom-entry-study2.xml" -H
"Content-Type: application/atom+xml" https://$HOSTNAME/dvn/api/data-deposit/
v1.1/swordv2/edit/study/doi:TEST/12345
```

### Delete a dataset

You must have the `DeleteDatasetDraft` permission (Contributor role or above such as Curator or Admin) on the dataset to delete it. Please note that if the dataset has never been published you will be able to delete it completely but if the dataset has already been published you will only be able to delete post-publication drafts, never a published version.

```
curl -u $API_TOKEN: -i -X DELETE https://$HOSTNAME/dvn/api/data-deposit/v1.1/
swordv2/edit/study/doi:TEST/12345
```



### Determine if a dataverse has been published

This API endpoint is the same as the “list datasets in a dataverse” endpoint documented above and the same permissions apply but it is documented here separately to point out that you can look for a boolean called `dataverseHasBeenReleased` to know if a dataverse has been released, which is required for publishing a dataset.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/collection/dataverse/$DATAVERSE_ALIAS
```

### Publish a dataverse

The `cat /dev/null` and `--data-binary @-` arguments are used to send zero-length content to the API, which is required by the upstream library to process the `In-Progress: false` header. You must have the `PublishDataverse` permission (Admin role) on the dataverse to publish it.

```
cat /dev/null | curl -u $API_TOKEN: -X POST -H "In-Progress: false"
--data-binary @- https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit/
dataverse/$DATAVERSE_ALIAS
```

### Publish a dataset

The `cat /dev/null` and `--data-binary @-` arguments are used to send zero-length content to the API, which is required by the upstream library to process the `In-Progress: false` header. You must have the `PublishDataset` permission (Curator or Admin role) on the dataset to publish it.

```
cat /dev/null | curl -u $API_TOKEN: -X POST -H "In-Progress: false"
--data-binary @- https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit/
study/doi:TEST/12345
```

## 3.8.5 Known issues

- Deleting a file from a published version (not a draft) creates a draft but doesn't delete the file: <https://github.com/IQSS/dataverse/issues/2464>
- The Service Document does not honor groups within groups: <https://github.com/IQSS/dataverse/issues/3056>
- Should see all the fields filled in for a dataset regardless of what the parent dataverse specifies: <https://github.com/IQSS/dataverse/issues/756>
- SWORD 2.0 Profile 6.4 “Retrieving the content” has not been implemented: <https://github.com/IQSS/dataverse/issues/183>
- Deaccessioning via API is not supported (it was in DVN 3.x): <https://github.com/IQSS/dataverse/issues/778>
- Let file metadata (i.e. description) be specified during zip upload: <https://github.com/IQSS/dataverse/issues/723>
- SWORD: Display of actual dterms xml element for equivalent of required field not found: <https://github.com/IQSS/dataverse/issues/1019>

### 3.8.6 Bug fixes in v1.1

- Fix `Abdera ArrayIndexOutOfBoundsException` with non-existent `atom-entry-study.xml` in SWORD jar (upstream ideally) <https://github.com/IQSS/dataverse/issues/893>

- Sword API: Can't create study when hidden characters are introduced in atom.xml <https://github.com/IQSS/dataverse/issues/894>

### 3.8.7 Client libraries

- Python: <https://github.com/swordapp/python-client-sword2>
- Java: <https://github.com/swordapp/JavaClient2.0>
- R: <https://github.com/IQSS/dataverse-client-r>
- Ruby: <https://github.com/swordapp/sword2ruby>
- PHP: <https://github.com/swordapp/swordappv2-php-library>

## 3.9 Client Libraries

Currently there are client libraries for Python, Javascript, R, and Java that can be used to develop against Dataverse APIs. We use the term “client library” on this page but “Dataverse SDK” (software development kit) is another way of describing these resources. They are designed to help developers express Dataverse concepts more easily in the languages listed below. For support on any of these client libraries, please consult each project’s README.

Because Dataverse is a SWORD server, additional client libraries exist for Java, Ruby, and PHP per the *SWORD API* page.

#### Contents:

- *Python*
- *Javascript*
- *R*
- *Java*

### 3.9.1 Python

There are two Python modules for interacting with Dataverse APIs.

`pyDataverse` had its initial release in 2019 and can be installed with `pip install pyDataverse`. The module is developed by Stefan Kasberger from AUSSDA - The Austrian Social Science Data Archive.

`dataverse-client-python` had its initial release in 2015. Robert Liebowitz created this library while at the Center for Open Science (COS) and the COS uses it to integrate the Open Science Framework (OSF) with Dataverse via an add-on which itself is open source and listed on the *Apps* page.

### 3.9.2 Javascript

<https://github.com/IQSS/dataverse-client-javascript> is the official Javascript package for Dataverse APIs. It can be found on npm at <https://www.npmjs.com/package/js-dataverse>

It was created and is maintained by The Agile Monkeys.

### 3.9.3 R

<https://github.com/IQSS/dataverse-client-r> is the official R package for Dataverse APIs. The latest release can be installed from CRAN.

It was created by Thomas Leeper whose dataverse can be found at <https://dataverse.harvard.edu/dataverse/leeper>

### 3.9.4 Java

<https://github.com/IQSS/dataverse-client-java> is the official Java library for Dataverse APIs.

Richard Adams from ResearchSpace created and maintains this library.

## 3.10 Building External Tools

External tools can provide additional features that are not part of Dataverse itself, such as data exploration. Thank you for your interest in building an external tool for Dataverse!

### Contents:

- *Introduction*
- *Examples of External Tools*
- *How External Tools Are Presented to Users*
- *Creating an External Tool Manifest*
  - *Examples of Manifests*
    - \* *External Tools for Files*
    - \* *External Tools for Datasets*
  - *Terminology*
  - *Reserved Words*
  - *Using Example Manifests to Get Started*
- *Testing Your External Tool*
- *Spreading the Word About Your External Tool*
  - *Adding Your Tool to the Inventory of External Tools*
  - *Demoing Your External Tool*
  - *Announcing Your External Tool*

### 3.10.1 Introduction

You can think of a external tool as a **glorified hyperlink** that opens a browser window in a new tab on some other website. The term “external” is used to indicate that the user has left the Dataverse web interface. For example, perhaps the user is looking at a dataset on <https://demo.dataverse.org> . They click “Explore” and are brought to <https://fabulousfiletool.com?fileId=42&siteUrl=http://demo.dataverse.org>

The “other website” (fabulousfiletool.com in the example above) is probably part of the same ecosystem of scholarly publishing that Dataverse itself participates in. Sometimes the other website runs entirely in the browser. Sometimes the other website is a full blown server side web application like Dataverse itself.

The possibilities for external tools are endless. Let’s look at some examples to get your creative juices flowing.

### 3.10.2 Examples of External Tools

Note: This is the same list that appears in the *External Tools* section of the Admin Guide.

Tool	Type	Scope	Description
TwoRavens	ex- plore	file	A system of interlocking statistical tools for data exploration, analysis, and meta-analysis: <a href="http://2ra.vn">http://2ra.vn</a> . See the <i>TwoRavens: Tabular Data Exploration</i> section of the User Guide for more information on TwoRavens from the user perspective and the <i>TwoRavens</i> section of the Installation Guide.
Data Explorer	ex- plore	file	A GUI which lists the variables in a tabular data file allowing searching, charting and cross tabulation analysis. See the README.md file at <a href="https://github.com/scholarsportal/Dataverse-Data-Explorer">https://github.com/scholarsportal/Dataverse-Data-Explorer</a> for the instructions on adding Data Explorer to your Dataverse; and the <i>Prerequisites</i> section of the Installation Guide for the instructions on how to set up <b>basic R configuration required</b> (specifically, Dataverse uses R to generate .prep metadata files that are needed to run Data Explorer).
Whole Tale	ex- plore	dataset	A platform for the creation of reproducible research packages that allows users to launch containerized interactive analysis environments based on popular tools such as Jupyter and RStudio. Using this integration, Dataverse users can launch Jupyter and RStudio environments to analyze published datasets. For more information, see the <a href="#">Whole Tale User Guide</a> .
File Pre-viewers	ex- plore	file	A set of tools that display the content of files - including audio, html, <a href="#">Hypothes.is</a> annotations, images, PDF, text, video - allowing them to be viewed without downloading. The previewers can be run directly from github.io, so the only required step is using the Dataverse API to register the ones you want to use. Documentation, including how to optionally brand the previewers, and an invitation to contribute through github are in the README.md file. <a href="https://github.com/QualitativeDataRepository/dataverse-previewers">https://github.com/QualitativeDataRepository/dataverse-previewers</a>
Data Curation Tool	con- fig- ure	file	A GUI for curating data by adding labels, groups, weights and other details to assist with informed reuse. See the README.md file at <a href="https://github.com/scholarsportal/Dataverse-Data-Curation-Tool">https://github.com/scholarsportal/Dataverse-Data-Curation-Tool</a> for the installation instructions.

### 3.10.3 How External Tools Are Presented to Users

In short, an external tool appears under an “Explore” or “Configure” button either on a dataset landing page or a file landing page. See also the *Testing External Tools* section of the Admin Guide for some perspective on how installations of Dataverse will expect to test your tool before announcing it to their users.

### 3.10.4 Creating an External Tool Manifest

External tools must be expressed in an external tool manifest file, a specific JSON format Dataverse requires. As the author of an external tool, you are expected to provide this JSON file and installation instructions on a web page for your tool.

## Examples of Manifests

Let's look at two examples of external tool manifests (one at the file level and one at the dataset level) before we dive into how they work.

### External Tools for Files

`fabulousFileTool.json` is a file level explore tool that operates on tabular files:

```
{
  "displayName": "Fabulous File Tool",
  "description": "Fabulous Fun for Files!",
  "scope": "file",
  "type": "explore",
  "toolUrl": "https://fabulousfiletool.com",
  "contentType": "text/tab-separated-values",
  "toolParameters": {
    "queryParameters": [
      {
        "fileid": "{fileId}"
      },
      {
        "key": "{apiToken}"
      }
    ]
  }
}
```

### External Tools for Datasets

`dynamicDatasetTool.json` is a dataset level explore tool:

```
{
  "displayName": "Dynamic Dataset Tool",
  "description": "Dazzles! Dizzying!",
  "scope": "dataset",
  "type": "explore",
  "toolUrl": "https://dynamicdatasettool.com/v2",
  "toolParameters": {
    "queryParameters": [
      {
        "PID": "{datasetPid}"
      },
      {
        "apiToken": "{apiToken}"
      }
    ]
  }
}
```

## Terminology

Term	Definition
external tool manifest	A <b>JSON file</b> that defines the URL constructed by Dataverse when users click “Explore” or “Configure” buttons. External tool makers are asked to host this JSON file on a website (no app store yet, sorry) and explain how to use install and use the tool. Examples include <code>fabulousFileTool.json</code> and <code>dynamicDatasetTool.json</code> as well as the real world examples above such as Data Explorer.
displayName	The <b>name</b> of the tool in the Dataverse web interface. For example, “Data Explorer”.
description	The <b>description</b> of the tool, which appears in a popup (for configure tools only) so the user who clicked the tool can learn about the tool before being redirected to the tool in a new tab in their browser. HTML is supported.
scope	Whether the external tool appears and operates at the <b>file</b> level or the <b>dataset</b> level. Note that a file level tool must also specify the type of file it operates on (see “contentType” below).
type	Whether the external tool is an <b>explore</b> tool or a <b>configure</b> tool. Configure tools require an API token because they make changes to data files (files within datasets). Configure tools are currently not supported at the dataset level (no “Configure” button appears in the GUI for datasets).
toolUrl	The <b>base URL</b> of the tool before query parameters are added.
contentType	File level tools operate on a specific <b>file type</b> (content type or MIME type such as “application/pdf”) and this must be specified. Dataset level tools do not use contentType.
toolParameters	<b>Query parameters</b> are supported and described below.
queryParameters	<b>Key/value combinations</b> that can be appended to the toolUrl. For example, once substitution takes place (described below) the user may be redirected to <code>https://fabulousfiletool.com?fileId=42&amp;siteUrl=http://demo.dataverse.org</code> .
query parameter keys	An <b>arbitrary string</b> to associate with a value that is populated with a reserved word (described below). As the author of the tool, you have control over what “key” you would like to be passed to your tool. For example, if you want to have your tool receive and operate on the query parameter “dataverseFileId=42” instead of just “fileId=42”, that’s fine.
query parameter values	A <b>mechanism for substituting reserved words with dynamic content</b> . For example, in your manifest file, you can use a reserved word (described below) such as <code>{fileId}</code> to pass a file’s database id to your tool in a query parameter. Your tool might receive this query parameter as “fileId=42”.
reserved words	A <b>set of strings surrounded by curly braces</b> such as <code>{fileId}</code> or <code>{datasetId}</code> that will be inserted into query parameters. See the table below for a complete list.

## Reserved Words

Reserved word	Status	Description
{siteUrl}	optional	The URL of the Dataverse installation from which the tool was launched. For example, <code>https://demo.dataverse.org</code> .
{fileId}	depends	The database ID of a file the user clicks “Explore” or “Configure” on. For example, 42. This reserved word is <b>required for file level tools</b> unless you use {filePid} instead.
{filePid}	depends	The Persistent ID (DOI or Handle) of a file the user clicks “Explore” or “Configure” on. For example, <code>doi:10.7910/DVN/TJCLKP/3VSTKY</code> . Note that not all installations of Dataverse have Persistent IDs (PIDs) enabled at the file level. This reserved word is <b>required for file level tools</b> unless you use {fileId} instead.
{apiToken}	optional	The Dataverse API token of the user launching the external tool, if available. Please note that API tokens should be treated with the same care as a password. For example, <code>f3465b0c-f830-4bc7-879f-06c0745a5a5c</code> .
{datasetId}	depends	The database ID of the dataset. For example, 42. This reserved word is <b>required for dataset level tools</b> unless you use {datasetPid} instead.
{datasetPid}	depends	The Persistent ID (DOI or Handle) of the dataset. For example, <code>doi:10.7910/DVN/TJCLKP</code> . This reserved word is <b>required for dataset level tools</b> unless you use {datasetId} instead.
{datasetVersion}	optional	The friendly version number (or <code>:draft</code> ) of the dataset version the file level tool is being launched from. For example, <code>1.0</code> or <code>:draft</code> .
{localeCode}	optional	The code for the language (“en” for English, “fr” for French, etc.) that user has selected from the language toggle in Dataverse. See also <i>Internationalization</i> .

## Using Example Manifests to Get Started

Again, you can use `fabulousFileTool.json` or `dynamicDatasetTool.json` as a starting point for your own manifest file.

### 3.10.5 Testing Your External Tool

As the author of an external tool, you are not expected to learn how to install and operate Dataverse. There’s a very good chance your tool can be added to a server Dataverse developers use for testing if you reach out on any of the channels listed under *Getting Help* in the Developer Guide.

By all means, if you’d like to install Dataverse yourself, a number of developer-centric options are available. For example, there’s a script to spin up Dataverse on EC2 at <https://github.com/IQSS/dataverse-sample-data>. The process for using curl to add your external tool to a Dataverse installation is documented under *Managing External Tools* in the Admin Guide.

### 3.10.6 Spreading the Word About Your External Tool

#### Adding Your Tool to the Inventory of External Tools

Once you’ve gotten your tool working, please make a pull request to update the list of tools above! You are also welcome to download `dataverse-external-tools.tsv`, add your tool to the TSV file, create and issue at <https://github.com/IQSS/dataverse/issues>, and then upload your TSV file there.

Unless your tool runs entirely in a browser, you may have integrated server-side software with Dataverse. If so, please double check that your software is listed in the *Integrations* section of the Admin Guide and if not, please open an issue or pull request to add it. Thanks!

If you've thought to yourself that there ought to be an app store for Dataverse external tools, you're not alone. Please see <https://github.com/IQSS/dataverse/issues/5688> :)

### Demoing Your External Tool

<https://demo.dataverse.org> is the place to play around with Dataverse and your tool can be included. Please email [support@dataverse.org](mailto:support@dataverse.org) to start the conversation about adding your tool. Additionally, you are welcome to open an issue at <https://github.com/IQSS/dataverse-ansible> which already includes a number of the tools listed above.

### Announcing Your External Tool

You are welcome to announce your external tool at <https://groups.google.com/forum/#!forum/dataverse-community>

If you're too shy, we'll do it for you. We'll probably tweet about it too. Thank you for your contribution to Dataverse!

## 3.11 Apps

The introduction of Dataverse APIs has fostered the development of a variety of software applications that are listed in the *Integrations*, *External Tools*, and *Reporting Tools* sections of the Admin Guide.

The apps below are open source and demonstrate how to use Dataverse APIs. Some of these apps are built on *Client Libraries* that are available for Dataverse APIs in Python, Javascript, R, and Java.

#### Contents:

- *Javascript*
  - *Data Explorer*
  - *Data Curation Tool*
  - *File Previewers*
  - *TwoRavens*
- *Python*
  - *dataverse-sample-data*
  - *Texas Digital Library dataverse-reports*
  - *OSF*
  - *GeoConnect*
  - *dataverse-metrics*
  - *Whole Tale*
  - *Archivematica*
- *Java*
  - *DVUploader*
  - *Dataverse for Android*
- *PHP*



### 3.11.1 Javascript

#### Data Explorer

Data Explorer is a GUI which lists the variables in a tabular data file allowing searching, charting and cross tabulation analysis.

<https://github.com/scholarsportal/Dataverse-Data-Explorer>

#### Data Curation Tool

Data Curation Tool is a GUI for curating data by adding labels, groups, weights and other details to assist with informed reuse.

<https://github.com/scholarsportal/Dataverse-Data-Curation-Tool>

#### File Previewers

File Previewers are tools that display the content of files - including audio, html, Hypothes.is annotations, images, PDF, text, video - allowing them to be viewed without downloading.

<https://github.com/QualitativeDataRepository/dataverse-previewers>

#### TwoRavens

TwoRavens is a system of interlocking statistical tools for data exploration, analysis, and meta-analysis.

<https://github.com/IQSS/TwoRavens>

### 3.11.2 Python

Please note that there are multiple Python modules for Dataverse APIs listed in the *Client Libraries* section.

#### **dataverse-sample-data**

dataverse-sample-data allows you to populate your Dataverse installation with sample data. It makes uses of pyData-verse, which is listed in the *Client Libraries* section.

<https://github.com/IQSS/dataverse-sample-data>

#### **Texas Digital Library dataverse-reports**

Dataverse Reports for Texas Digital Library generates and emails statistical reports for an installation of Dataverse using the native API and database queries.

<https://github.com/TexasDigitalLibrary/dataverse-reports>

### OSF

OSF allows you to view, download, and upload files to and from a Dataverse dataset from an Open Science Framework (OSF) project.

<https://github.com/CenterForOpenScience/osf.io/tree/develop/addons/dataverse>

### GeoConnect

GeoConnect allows Dataverse files to be visualized on <http://worldmap.harvard.edu> with the “Explore” button. Read more about it in the *WorldMap: Geospatial Data Exploration* section of the User Guide.

<https://github.com/IQSS/geoconnect>

### dataverse-metrics

dataverse-metrics aggregates and visualizes metrics across multiple Dataverse installations but can also be used with a single installation

<https://github.com/IQSS/dataverse-metrics>

### Whole Tale

Whole Tale enables researchers to analyze data using popular tools including Jupyter and RStudio with the ultimate goal of supporting publishing of reproducible research packages.

[https://github.com/whole-tale/girder\\_wholetale/tree/v0.7/server/lib/dataverse](https://github.com/whole-tale/girder_wholetale/tree/v0.7/server/lib/dataverse)

### Archivematica

Archivematica is an integrated suite of open-source tools for processing digital objects for long-term preservation.

<https://github.com/artefactual/archivematica/tree/v1.9.2/src/MCPClient/lib/clientScripts>

## 3.11.3 Java

Please note that there is a Java library for Dataverse APIs listed in the *Client Libraries* section.

### DVUploader

The open-source DVUploader tool is a stand-alone command-line Java application that uses the Dataverse API to upload files to a specified Dataset. Files can be specified by name, or the DVUploader can upload all files in a directory or recursively from a directory tree. The DVUploader can also verify that uploaded files match their local sources by comparing the local and remote fixity checksums. Source code, release 1.0.0- jar file, and documentation are available on GitHub. DVUploader’s creation was supported by the Texas Digital Library.

<https://github.com/IQSS/dataverse-uploader>

### Dataverse for Android

Dataverse for Android makes use of Dataverse’s Search API.

<https://github.com/IQSS/dataverse-android>

### 3.11.4 PHP

#### OJS

The Open Journal Systems (OJS) Dataverse Plugin adds data sharing and preservation to the OJS publication process.

[https://github.com/pkp/ojs/tree/ojs-stable-2\\_4\\_8/plugins/generic/dataverse](https://github.com/pkp/ojs/tree/ojs-stable-2_4_8/plugins/generic/dataverse)

## 3.12 Frequently Asked Questions

APIs are less intuitive than graphical user interfaces (GUIs) so questions are expected!

#### Contents:

- *What is an API?*
- *What Are Common Use Cases for Dataverse APIs?*
- *Where Can I Find Examples of Using Dataverse APIs?*
- *When Should I Use the Native API vs. the SWORD API?*
- *To Operate on a Dataset Should I Use Its DOI (or Handle) or Its Database ID?*
- *Where is the Comprehensive List of All API Functionality?*
- *Is There a Changelog of API Functionality That Has Been Added Over Time?*
- *What Functionality is GUI Only and Not Available Via API*
- *Why Are the Return Values (HTTP Status Codes) Not Documented?*
- *What If My Question Is Not Answered Here?*

### 3.12.1 What is an API?

See “What is an API?” in the *Introduction* section.

### 3.12.2 What Are Common Use Cases for Dataverse APIs?

See the *Getting Started with APIs* section for common use cases for researchers and curators. Other types of API users should find starting points at *Types of Dataverse API Users*.

### 3.12.3 Where Can I Find Examples of Using Dataverse APIs?

See the *Getting Started with APIs* section links to examples using curl.

For examples in Javascript, Python, R, and Java, and PHP, see the *Apps* and *Client Libraries* sections.

### 3.12.4 When Should I Use the Native API vs. the SWORD API?

The *SWORD API* is based on a standard, works fine, and is fully supported, but much more development effort has been going into the *Native API*, which is not based on a standard. It is specific to Dataverse.

SWORD uses XML. The Native API uses JSON.

SWORD only supports a dozen or so operations. The Native API supports many more.

### 3.12.5 To Operate on a Dataset Should I Use Its DOI (or Handle) or Its Database ID?

It is fine to target a datasets using either its Persistent ID (PID such as DOI or Handle) or its database id.

Here's an example from *Publish a Dataset* of targeting a dataset using its DOI:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -X POST "https://demo.
↪dataverse.org/api/datasets/:persistentId/actions/:publish?persistentId=doi:10.5072/
↪FK2/J8SJZB&type=major"
```

You can target the same dataset with its database ID (“42” in the example below), like this:

```
curl -H X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -X POST "https://demo.
↪dataverse.org/api/datasets/42/actions/:publish?type=major"
```

Note that when multiple query parameters are used (such as `persistentId` and `type` above) there is a question mark (?) before the first query parameter and ampersands (&) before each of the subsequent query parameters. Also, & has special meaning in Unix shells such as Bash so you must put quotes around the entire URL.

### 3.12.6 Where is the Comprehensive List of All API Functionality?

There are so many Dataverse APIs that a single page in this guide would probably be overwhelming. See *Lists of Dataverse APIs* for links to various pages.

It is possible to get a complete list of API functionality in Swagger/OpenAPI format if you deploy Dataverse to Payara 5+. For details, see <https://github.com/IQSS/dataverse/issues/5794>

### 3.12.7 Is There a Changelog of API Functionality That Has Been Added Over Time?

No, but there probably should be. If you have suggestions for how it should look, please create an issue at <https://github.com/IQSS/dataverse/issues>

### 3.12.8 What Functionality is GUI Only and Not Available Via API

The following tasks cannot currently be automated via API because no API exists for them. The web interface should be used instead for these GUI-only features:

- Setting a logo image, URL, and tagline when creating a dataverse.
- Editing properties of an existing dataverse.
- Set “Enable Access Request” for Terms of Use: <https://groups.google.com/d/msg/dataverse-community/oKdesT9rFGc/qM6wrsnnBAAJ>
- Downloading a guestbook.
- Set `guestbook_id` for a dataset: <https://groups.google.com/d/msg/dataverse-community/oKdesT9rFGc/qM6wrsnnBAAJ>
- Filling out a guestbook. See also [https://groups.google.com/d/msg/dataverse-dev/G9FNGP\\_bT0w/dgE2Fk4iBQAJ](https://groups.google.com/d/msg/dataverse-dev/G9FNGP_bT0w/dgE2Fk4iBQAJ)

- Seeing why a file failed ingest.
- Dataset templates.
- Deaccessioning datasets.

If you would like APIs for any of the features above, please open a GitHub issue at <https://github.com/IQSS/dataverse/issues>

You are also welcome to open an issue to add to the list above. Or you are welcome to make a pull request. Please see the *Writing Documentation* section of the Developer Guide for instructions.

### 3.12.9 Why Are the Return Values (HTTP Status Codes) Not Documented?

They should be. Please consider making a pull request to help. The *Writing Documentation* section of the Developer Guide should help you get started. *Create a Dataverse* has an example you can follow or you can come up with a better way.

### 3.12.10 What If My Question Is Not Answered Here?

Please ask! For information on where to ask, please see *Getting Help*.



## INSTALLATION GUIDE

### Contents:

## 4.1 Introduction

Welcome! Thanks for installing Dataverse!

### Contents:

- *Quick Links*
- *Intended Audience*
- *Related Guides*
- *Getting Help*
- *Improving this Guide*

### 4.1.1 Quick Links

If you are installing Dataverse for the first time, please proceed to the *Preparation* section.

Jump ahead to *Configuration* or *Upgrading* for an existing Dataverse installation.

### 4.1.2 Intended Audience

This guide is intended primarily for sysadmins who are installing, configuring, and upgrading Dataverse.

Sysadmins are expected to be comfortable using standard Linux commands, issuing `curl` commands, and running SQL scripts.

### 4.1.3 Related Guides

Many “admin” functions can be performed by Dataverse users themselves (non-superusers) as documented in the *User Guide* and that guide is a good introduction to the features of Dataverse from an end user perspective.

If you are a sysadmin who likes to code, you may find the *API Guide* useful, and you may want to consider improving the installation script or hacking on the community-lead configuration management options mentioned in the *Preparation* section. If you **really** like to code and want to help with the Dataverse code, please check out the *Developer Guide*!

### 4.1.4 Getting Help

To get help installing or configuring Dataverse, please try one or more of:

- posting to the [dataverse-community](#) Google Group.
- asking at <http://chat.dataverse.org> (#dataverse on the freenode IRC network)
- emailing [support@dataverse.org](mailto:support@dataverse.org) to open a private ticket at <https://help.hmdc.harvard.edu>

### 4.1.5 Improving this Guide

If you spot a typo in this guide or would like to suggest an improvement, please find the appropriate file in <https://github.com/IQSS/dataverse/tree/develop/doc/sphinx-guides/source/installation> and send a pull request as explained in the *Writing Documentation* section of the Developer Guide. You are also welcome to simply open an issue at <https://github.com/IQSS/dataverse/issues> to describe the problem with this guide.

Next is the *Preparation* section.

## 4.2 Preparation

```
> "What are you preparing? You're always preparing! Just go!" -- Spaceballs
```

We'll try to get you up and running as quickly as possible, but we thought you might like to hear about your options. :)

### Contents:

- *Choose Your Own Installation Adventure*
  - *Vagrant (for Testing Only)*
  - *Pilot Installation*
  - *Advanced Installation*
- *Architecture and Components*
  - *Required Components*
  - *Optional Components*
- *System Requirements*
  - *Hardware Requirements*
  - *Software Requirements*
- *Decisions to Make*
- *Next Steps*



## 4.2.1 Choose Your Own Installation Adventure

### Vagrant (for Testing Only)

If you are looking to simply kick the tires on installing Dataverse and are familiar with Vagrant, you are welcome to read through the “Vagrant” section of the *Tools* section of the Developer Guide. Checking out a tagged release is recommended rather than running `vagrant up` on unreleased code.

### Pilot Installation

Vagrant is not a bad way for a sysadmin to get a quick sense of how an application like Dataverse is put together in a sandbox (a virtual machine running on a laptop for example), but to allow end users to start playing with Dataverse, you’ll need to install Dataverse on a server.

Installing Dataverse involves some system configuration followed by executing an installation script that will guide you through the installation process as described in *Installation*, but reading about the *Architecture and Components* of Dataverse is recommended first.

### Advanced Installation

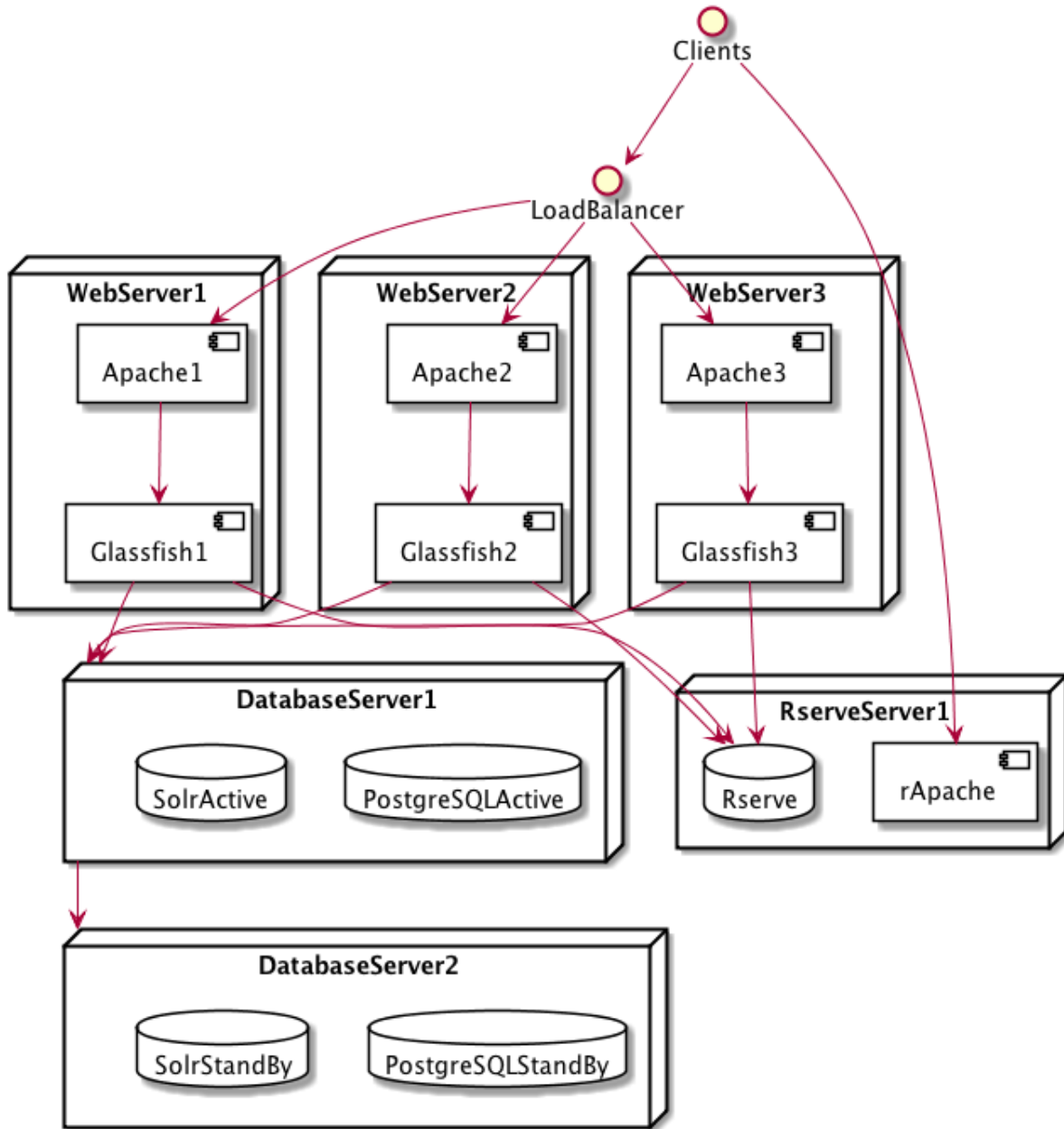
There are some community-lead projects to use configuration management tools such as Ansible and Puppet to automate Dataverse installation and configuration, but support for these solutions is limited to what the Dataverse community can offer as described in each project’s webpage:

- <https://github.com/IQSS/dataverse-ansible>
- <https://github.com/IQSS/dataverse-puppet>

(Please note that the “dataverse-ansible” repo is used in a script that allows Dataverse to be installed on Amazon Web Services (AWS) from arbitrary GitHub branches as described in the *Deployment* section of the Developer Guide.)

The Dataverse development team is happy to “bless” additional community efforts along these lines (i.e. Docker, Chef, Salt, etc.) by creating a repo under <https://github.com/IQSS> and managing team access.

Dataverse permits a fair amount of flexibility in where you choose to install the various components. The diagram below shows a load balancer, multiple proxies and web servers, redundant database servers, and offloading of potentially resource intensive work to a separate server.



A setup such as this is advanced enough to be considered out of scope for this guide (apart from a stub in the [Advanced Installation](#) section) but you are welcome to ask questions about similar configurations via the support channels listed in the [Introduction](#).

## 4.2.2 Architecture and Components

Dataverse is a Java Enterprise Edition (EE) web application that is shipped as a war (web archive) file.

### Required Components

When planning your installation you should be aware of the following components of the Dataverse architecture:

- Linux: RHEL/CentOS is highly recommended since all development and QA happens on this distribution.
- Glassfish: a Java EE application server to which the Dataverse application (war file) is to be deployed.
- PostgreSQL: a relational database.
- Solr: a search engine. A Dataverse-specific schema is provided.
- SMTP server: for sending mail for password resets and other notifications.
- Persistent identifier service: DOI and Handle support are provided. Production use requires a registered DOI or Handle.net authority.

## Optional Components

There are a number of optional components you may choose to install or configure, including:

- External Tools: Third party tools for data exploration can be added to Dataverse by following the instructions in the *External Tools* section of the Admin Guide.
- R, rApache, Zelig, and TwoRavens: *TwoRavens: Tabular Data Exploration* describes the feature and *TwoRavens* describes how to install these components. *External Tools* explains how third-party tools like TwoRavens can be added to Dataverse.
- Dropbox integration *dataverse.dropbox.key*: for uploading files from the Dropbox API.
- Apache: a web server that can “reverse proxy” Glassfish applications and rewrite HTTP traffic.
- Shibboleth: an authentication system described in *Shibboleth*. Its use with Dataverse requires Apache.
- OAuth2: an authentication system described in *OAuth Login: ORCID, GitHub, Google*.
- Geoconnect: a system that allows users to create maps from geospatial files, described in *Geoconnect*.

See also the *Integrations* section of the Admin Guide.

## 4.2.3 System Requirements

### Hardware Requirements

A basic installation of Dataverse runs fine on modest hardware. For example, as of this writing the test installation at <http://phoenix.dataverse.org> is backed by a single virtual machine with two 2.8 GHz processors, 8 GB of RAM and 50 GB of disk.

In contrast, before we moved it to the Amazon Cloud, the production installation at <https://dataverse.harvard.edu> was backed by six servers with two Intel Xeon 2.53 Ghz CPUs and either 48 or 64 GB of RAM. The three servers with 48 GB of RAM run were web frontends running Glassfish and Apache and were load balanced by a hardware device. The remaining three servers with 64 GB of RAM were the primary and backup database servers and a server dedicated to running Rserve. Multiple TB of storage were mounted from a SAN via NFS.

Currently, Harvard Dataverse is served by four AWS server nodes: two “m4.4xlarge” instances (64GB/16 vCPU) as web frontends, one 32GB/8 vCPU (“m4.2xlarge”) instance for the Solr search engine, and one 16GB/4 vCPU (“m4.xlarge”) instance for R and TwoRavens. The PostgreSQL database is served by Amazon RDS, and physical files are stored on Amazon S3.

The Dataverse installation script will attempt to give Glassfish the right amount of RAM based on your system.

Experimentation and testing with various hardware configurations is encouraged, or course, but do reach out as explained in the *Introduction* as needed for assistance.

## Software Requirements

See *Architecture and Components* for an overview of required and optional components. The *Prerequisites* section is oriented toward installing the software necessary to successfully run the Dataverse installation script. Pages on optional components contain more detail of software requirements for each component.

Clients are expected to be running a relatively modern browser.

### 4.2.4 Decisions to Make

Here are some questions to keep in the back of your mind as you test and move into production:

- How much storage do I need?
- Which features do I want based on *Architecture and Components*?
- How do I want my users to log in to Dataverse? With local accounts? With Shibboleth/SAML? With OAuth providers such as ORCID, GitHub, or Google?
- Do I want to run Glassfish on the standard web ports (80 and 443) or do I want to “front” Glassfish with a proxy such as Apache or nginx? See “Network Ports” in the *Configuration* section.
- How many points of failure am I willing to tolerate? How much complexity do I want?
- How much does it cost to subscribe to a service to create persistent identifiers such as DOIs or handles?

### 4.2.5 Next Steps

Proceed to the *Prerequisites* section which will help you get ready to run the Dataverse installation script.

## 4.3 Prerequisites

Before running the Dataverse installation script, you must install and configure Linux, Java, Glassfish, PostgreSQL, Solr, and jq. The other software listed below is optional but can provide useful features.

After following all the steps below, you can proceed to the *Installation* section.

You **may** find it helpful to look at how the configuration is done automatically by various tools such as Vagrant, Puppet, or Ansible. See the *Preparation* section for pointers on diving into these scripts.

#### Contents:

- *Linux*
- *Java*
  - *Installing Java*
- *Glassfish*
  - *Installing Glassfish*
  - *Launching Glassfish on system boot*
- *PostgreSQL*
  - *Installing PostgreSQL*

- *Configuring Database Access for the Dataverse Application (and the Dataverse Installer)*
- *Solr*
  - *Installing Solr*
  - *Solr Init Script*
  - *Securing Solr*
- *jq*
  - *Installing jq*
- *ImageMagick*
  - *Installing and configuring ImageMagick*
- *R*
  - *Installing R*
  - *Installing the required R libraries*
  - *Rserve*
- *Counter Processor*
  - *Installing Counter Processor*
  - *Installing Counter Processor Python Requirements*
- *Next Steps*

### 4.3.1 Linux

We assume you plan to run Dataverse on Linux and we recommend RHEL/CentOS, which is the Linux distribution tested by the Dataverse development team. Please be aware that while el7 (RHEL/CentOS 7) is the recommended platform, the steps below were originally written for el6 and may need to be updated (please feel free to make a pull request!).

### 4.3.2 Java

Dataverse requires Java SE 8 (8u74/JDK 1.8.0u74 or higher).

#### Installing Java

Dataverse should run fine with only the Java Runtime Environment (JRE) installed, but installing the Java Development Kit (JDK) is recommended so that useful tools for troubleshooting production environments are available. We recommend using Oracle JDK or OpenJDK.

The Oracle JDK can be downloaded from <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

On a RHEL/CentOS, install OpenJDK (devel version) using yum:

```
# yum install java-1.8.0-openjdk-devel
```

If you have multiple versions of Java installed, Java 8 should be the default when `java` is invoked from the command line. You can test this by running `java -version`.

On RHEL/CentOS you can make Java 8 the default with the `alternatives` command, having it prompt you to select the version of Java from a list:

```
# alternatives --config java
```

If you don't want to be prompted, here is an example of the non-interactive invocation:

```
# alternatives --set java /usr/lib/jvm/jre-1.8.0-openjdk.x86_64/bin/java
```

### 4.3.3 Glassfish

Glassfish Version 4.1 is required. There are known issues with newer versions of the Glassfish 4.x series so it should be avoided. For details, see <https://github.com/IQSS/dataverse/issues/2628> . The issue we are using the track support for Glassfish 5 is <https://github.com/IQSS/dataverse/issues/4248> .

#### Installing Glassfish

**Note:** The Dataverse installer need not be run as root, and it is recommended that Glassfish not run as root either. We suggest the creation of a glassfish service account for this purpose.

- Download and install Glassfish (installed in `/usr/local/glassfish4` in the example commands below):

```
# wget http://dlc-cdn.sun.com/glassfish/4.1/release/glassfish-4.1.zip
# unzip glassfish-4.1.zip
# mv glassfish4 /usr/local
```

If you intend to install and run Glassfish under a service account (and we hope you do), `chown -R` the Glassfish hierarchy to root to protect it but give the service account access to the below directories:

- Set service account permissions:

```
# chown -R root:root /usr/local/glassfish4
# chown glassfish /usr/local/glassfish4/glassfish/lib
# chown -R glassfish:glassfish /usr/local/glassfish4/glassfish/domains/domain1
```

After installation, you may `chown` the `lib/` directory back to root; the installer only needs write access to copy the JDBC driver into that directory.

Once Glassfish is installed, you'll need a newer version of the Weld library (v2.2.10.SP1) to fix a serious issue in the library supplied with Glassfish 4.1 (see <https://github.com/IQSS/dataverse/issues/647> for details). If you plan to front Glassfish with Apache you must also patch Grizzly as explained in the *Shibboleth* section.

- Remove the stock Weld jar; download Weld v2.2.10.SP1 and install it in the modules folder:

```
# cd /usr/local/glassfish4/glassfish/modules
# rm weld-osgi-bundle.jar
# wget http://central.maven.org/maven2/org/jboss/weld/weld-osgi-bundle/2.2.10.SP1/
↪weld-osgi-bundle-2.2.10.SP1-glassfish4.jar
```

- Change from `-client` to `-server` under `<jvm-options>-client</jvm-options>`:

```
# vim /usr/local/glassfish4/glassfish/domains/domain1/config/domain.xml
```

This recommendation comes from <http://www.c2b2.co.uk/middleware-blog/glassfish-4-performance-tuning-monitoring-and-troubleshooting> among other places.

- Start Glassfish and verify the Weld version:

```
# /usr/local/glassfish4/bin/asadmin start-domain
# /usr/local/glassfish4/bin/asadmin osgi lb | grep 'Weld OSGi Bundle'
```

The Certificate Authority (CA) certificate bundle file from Glassfish contains certs that expired in August 2018, causing problems with ORCID login.

- The actual expiration date is August 22, 2018, which you can see with the following command:

```
# keytool -list -v -keystore /usr/local/glassfish4/glassfish/domains/domain1/
↳config/cacerts.jks
```

- Overwrite Glassfish’s CA certs file with the file that ships with the operating system and restart Glassfish:

```
# cp /etc/pki/ca-trust/extracted/java/cacerts /usr/local/glassfish4/glassfish/
↳domains/domain1/config/cacerts.jks
# /usr/local/glassfish4/bin/asadmin stop-domain
# /usr/local/glassfish4/bin/asadmin start-domain
```

## Launching Glassfish on system boot

The Dataverse installation script will start Glassfish if necessary, but you may find the following scripts helpful to launch Glassfish start automatically on boot.

- This `Systemd` file may be serve as a reference for systems using Systemd (such as RHEL/CentOS 7 or Ubuntu 16+)
- This `init` script may be useful for RHEL/CentOS 6 or Ubuntu >= 14 if you’re using a Glassfish service account, or
- This `Glassfish init` script may be helpful if you’re just going to run Glassfish as root.

It is not necessary for Glassfish to be running before you execute the Dataverse installation script; it will start Glassfish for you.

Please note that you must run Glassfish in an English locale. If you are using something like `LANG=de_DE.UTF-8`, ingest of tabular data will fail with the message “RoundRoutines:decimal separator no in right place”.

Also note that Glassfish may utilize more than the default number of file descriptors, especially when running batch jobs such as harvesting. We have increased ours by adding `ulimit -n 32768` to our glassfish init script. On operating systems which use systemd such as RHEL or CentOS 7, file descriptor limits may be increased by adding a line like `LimitNOFILE=32768` to the systemd unit file. You may adjust the file descriptor limits on running processes by using the `prlimit` utility:

```
# sudo prlimit --pid pid --nofile=32768:32768
```

## 4.3.4 PostgreSQL

### Installing PostgreSQL

Version 9.6 is strongly recommended because it is the version developers and QA test with:

```
# yum install -y https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_
↳64/pgdg-centos96-9.6-3.noarch.rpm
# yum makecache fast
# yum install -y postgresql96-server
```

```
# /usr/pgsql-9.6/bin/postgresql96-setup initdb
# /usr/bin/systemctl start postgresql-9.6
# /usr/bin/systemctl enable postgresql-9.6
```

Note that the steps above are specific to RHEL/CentOS 7. For RHEL/CentOS 6 use:

```
# yum install -y https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-6-x86_
→64/pgdg-centos96-9.6-3.noarch.rpm
# yum makecache fast
# yum install -y postgresql96-server
# service postgresql-9.6 initdb
# service postgresql-9.6 start
```

### Configuring Database Access for the Dataverse Application (and the Dataverse Installer)

- The application and the installer script will be connecting to PostgreSQL over TCP/IP, using password authentication. In this section we explain how to configure PostgreSQL to accept these connections.
- If PostgreSQL is running on the same server as Glassfish, find the localhost (127.0.0.1) entry that's already in the `pg_hba.conf` and modify it to look like this:

```
host all all 127.0.0.1/32 md5
```

Once you are done with the prerequisites and run the installer script (documented here: [Installation](#)) it will ask you to enter the address of the Postgres server. Simply accept the default value `127.0.0.1` there.

- The Dataverse installer script will need to connect to PostgreSQL **as the admin user**, in order to create and set up the database that the Dataverse will be using. If for whatever reason it is failing to connect (for example, if you don't know/remember what your Postgres admin password is), you may choose to temporarily disable all the access restrictions on localhost connections, by changing the above line to:

```
host all all 127.0.0.1/32 trust
```

Note that this rule opens access to the database server **via localhost only**. Still, in a production environment, this may constitute a security risk. So you will likely want to change it back to "md5" once the installer has finished.

- If the Dataverse application is running on a different server, you will need to add a new entry to the `pg_hba.conf` granting it access by its network address:

```
host all all [ADDRESS] 255.255.255.255 md5
```

Where `[ADDRESS]` is the numeric IP address of the Glassfish server. Enter this address when the installer asks for the PostgreSQL server address.

- In some distributions, PostgreSQL is pre-configured so that it doesn't accept network connections at all. Check that the `listen_address` line in the configuration file `postgresql.conf` is not commented out and looks like this:

```
listen_addresses='*'
```

The file `postgresql.conf` will be located in the same directory as the `pg_hba.conf` above.

- **Important: PostgreSQL must be restarted** for the configuration changes to take effect! On RHEL/CentOS 7 and similar (provided you installed Postgres as instructed above):



```
# systemctl restart postgresql-9.6
```

On MacOS X a “Reload Configuration” icon is usually supplied in the PostgreSQL application folder. Or you could look up the process id of the PostgreSQL postmaster process, and send it the SIGHUP signal:

```
kill -1 PROCESS_ID
```

### 4.3.5 Solr

The Dataverse search index is powered by Solr.

#### Installing Solr

You should not run Solr as root. Create a user called `solr` and a directory to install Solr into:

```
useradd solr
mkdir /usr/local/solr
chown solr:solr /usr/local/solr
```

Become the `solr` user and then download and configure Solr:

```
su - solr
cd /usr/local/solr
wget https://archive.apache.org/dist/lucene/solr/7.3.1/solr-7.3.1.tgz
tar xvzf solr-7.3.1.tgz
cd solr-7.3.1
cp -r server/solr/configsets/_default server/solr/collection1
```

You should already have a “`dvinstall.zip`” file that you downloaded from <https://github.com/IQSS/dataverse/releases>. Unzip it into `/tmp`. Then copy the files into place:

```
cp /tmp/dvinstall/schema*.xml /usr/local/solr/solr-7.3.1/server/solr/collection1/conf
cp /tmp/dvinstall/solrconfig.xml /usr/local/solr/solr-7.3.1/server/solr/collection1/
↪conf
```

Note: Dataverse has customized Solr to boost results that come from certain indexed elements inside Dataverse, for example prioritizing results from Dataverses over Datasets. If you would like to remove this, edit your `solrconfig.xml` and remove the `<str name="qf">` element and its contents. If you have ideas about how this boosting could be improved, feel free to contact us through our Google Group <https://groups.google.com/forum/#!forum/dataverse-dev>.

Dataverse requires a change to the `jetty.xml` file that ships with Solr. Edit `/usr/local/solr/solr-7.3.1/server/etc/jetty.xml`, increasing `requestHeaderSize` from 8192 to 102400

Solr will warn about needing to increase the number of file descriptors and max processes in a production environment but will still run with defaults. We have increased these values to the recommended levels by adding `ulimit -n 65000` to the init script, and the following to `/etc/security/limits.conf`:

```
solr soft nproc 65000
solr hard nproc 65000
solr soft nofile 65000
solr hard nofile 65000
```

On operating systems which use systemd such as RHEL or CentOS 7, you may then add a line like `LimitNOFILE=65000` for the number of open file descriptors and a line with `LimitNPROC=65000` for the max processes to the systemd unit file, or adjust the limits on a running process using the `prlimit` tool:

```
# sudo prlimit --pid pid --nofile=65000:65000
```

Solr launches asynchronously and attempts to use the `lsof` binary to watch for its own availability. Installation of this package isn't required but will prevent a warning in the log at startup:

```
# yum install lsof
```

Finally, you need to tell Solr to create the core “collection1” on startup:

```
echo "name=collection1" > /usr/local/solr/solr-7.3.1/server/solr/collection1/core.properties
```

### Solr Init Script

Please choose the right option for your underlying Linux operating system. It will not be necessary to execute both!

For systems running systemd (like CentOS/RedHat since 7, Debian since 9, Ubuntu since 15.04), as root, download `solr.service` and place it in `/tmp`. Then start Solr and configure it to start at boot with the following commands:

```
cp /tmp/solr.service /etc/systemd/system
systemctl daemon-reload
systemctl start solr.service
systemctl enable solr.service
```

For systems using `init.d` (like CentOS 6), download this `Solr init script` and place it in `/tmp`. Then start Solr and configure it to start at boot with the following commands:

```
cp /tmp/solr /etc/init.d
service start solr
chkconfig solr on
```

### Securing Solr

Solr must be firewalled off from all hosts except the server(s) running Dataverse. Otherwise, any host that can reach the Solr port (8983 by default) can add or delete data, search unpublished data, and even reconfigure Solr. For more information, please see [https://lucene.apache.org/solr/guide/7\\_2/securing-solr.html](https://lucene.apache.org/solr/guide/7_2/securing-solr.html)

## 4.3.6 jq

### Installing jq

`jq` is a command line tool for parsing JSON output that is used by the Dataverse installation script. It is available in the EPEL repository:

```
# yum install epel-release
# yum install jq
```

or you may install it manually:

```
# cd /usr/bin
# wget http://stedolan.github.io/jq/download/linux64/jq
# chmod +x jq
# jq --version
```

### 4.3.7 ImageMagick

Dataverse uses [ImageMagick](#) to generate thumbnail previews of PDF files. This is an optional component, meaning that if you don't have ImageMagick installed, there will be no thumbnails for PDF files, in the search results and on the dataset pages; but everything else will be working. (Thumbnail previews for non-PDF image files are generated using standard Java libraries and do not require any special installation steps).

#### Installing and configuring ImageMagick

On a Red Hat and similar Linux distributions, you can install ImageMagick with something like:

```
# yum install ImageMagick
```

(most RedHat systems will have it pre-installed). When installed using standard `yum` mechanism, above, the executable for the ImageMagick convert utility will be located at `/usr/bin/convert`. No further configuration steps will then be required.

On MacOS you can compile ImageMagick from sources, or use one of the popular installation frameworks, such as `brew`.

If the installed location of the convert executable is different from `/usr/bin/convert`, you will also need to specify it in your Glassfish configuration using the JVM option, below. For example:

```
<jvm-options>-Ddataverse.path.imagemagick.convert=/opt/local/bin/convert</jvm-options>
```

(see the [Configuration](#) section for more information on the JVM options)

### 4.3.8 R

Dataverse uses [R](#) to handle tabular data files. The instructions below describe a **minimal** R installation. It will allow you to ingest R (.RData) files as tabular data; to export tabular data as .RData files; and to run [Data Explorer](#) (specifically, R is used to generate .prep metadata files that Data Explorer uses). R can be considered an optional component, meaning that if you don't have R installed, you will still be able to run and use Dataverse - but the functionality specific to tabular data mentioned above will not be available to your users. **Note** that if you choose to also install [TwoRavens](#), it will require some extra R components and libraries. Please consult the instructions in the [TwoRavens](#) section of the Installation Guide.

#### Installing R

Can be installed with yum:

```
yum install R-core R-core-devel
```

EPEL distribution is strongly recommended. The version of R currently available from epel6 and epel7 is 3.5; it has been tested and is known to work on RedHat and CentOS versions 6 and 7.

If yum isn't configured to use EPEL repositories ( <https://fedoraproject.org/wiki/EPEL> ):

RHEL/CentOS users can install the RPM epel-release. For RHEL/CentOS 7:

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

RHEL/CentOS users can install the RPM epel-release. For RHEL/CentOS 6:

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-6.noarch.rpm
```

RHEL users will want to log in to their organization's respective RHN interface, find the particular machine in question and:

- click on “Subscribed Channels: Alter Channel Subscriptions”
- enable EPEL, Server Extras, Server Optional

### Installing the required R libraries

The following R packages (libraries) are required:

```
R2HTML  
rjson  
DescTools  
Rserve  
haven
```

Install them following the normal R package installation procedures. For example, with the following R commands:

```
install.packages("R2HTML", repos="https://cloud.r-project.org/", lib="/usr/lib64/R/  
↪library" )  
install.packages("rjson", repos="https://cloud.r-project.org/", lib="/usr/lib64/R/  
↪library" )  
install.packages("DescTools", repos="https://cloud.r-project.org/", lib="/usr/lib64/R/  
↪library" )  
install.packages("Rserve", repos="https://cloud.r-project.org/", lib="/usr/lib64/R/  
↪library" )  
install.packages("haven", repos="https://cloud.r-project.org/", lib="/usr/lib64/R/  
↪library" )
```

### Rserve

Dataverse uses [Rserve](#) to communicate to R. Rserve is installed as a library package, as described in the step above. It runs as a daemon process on the server, accepting network connections on a dedicated port. This requires some extra configuration and we provide a script (`scripts/r/rserve/rserve-setup.sh`) for setting it up. Run the script as follows (as root):

```
cd <DATAVERSE SOURCE TREE>/scripts/r/rserve  
./rserve-setup.sh
```

The setup script will create a system user `rserve` that will run the daemon process. It will install the startup script for the daemon (`/etc/init.d/rserve`), so that it gets started automatically when the system boots. This is an `init.d`-style startup file. If this is a RedHat/CentOS 7 system, you may want to use the `rserve.service` systemd unit file instead. Copy it into the `/usr/lib/systemd/system/` directory, then:

```
# systemctl daemon-reload  
# systemctl enable rserve  
# systemctl start rserve
```

Note that the setup will also set the Rserve password to “rserve”. Rserve daemon runs under a non-privileged user id, so there’s not much potential for security damage through unauthorized access. It is however still a good idea **to change the password**. The password is specified in /etc/Rserv.pwd. You can consult [Rserve documentation](#) for more information on password encryption and access security.

You should already have the following 4 JVM options added to your domain.xml by the Dataverse installer:

```
<jvm-options>-Ddataverse.rserve.host=localhost</jvm-options>
<jvm-options>-Ddataverse.rserve.port=6311</jvm-options>
<jvm-options>-Ddataverse.rserve.user=rserve</jvm-options>
<jvm-options>-Ddataverse.rserve.password=rserve</jvm-options>
```

If you have changed the password, make sure it is correctly specified in the dataverse.rserve.password option above. If Rserve is running on a host that’s different from your Dataverse server, change the dataverse.rserve.host option above as well (and make sure the port 6311 on the Rserve host is not firewalled from your Dataverse host).

### 4.3.9 Counter Processor

Counter Processor is required to enable Make Data Count metrics in Dataverse. See the *Make Data Count* section of the Admin Guide for a description of this feature. Counter Processor is open source and we will be downloading it from <https://github.com/CDLUC3/counter-processor>

#### Installing Counter Processor

Counter Processor has only been tested on el7 (see “Linux” above). Please note that a scripted installation using Ansible is mentioned in the *Make Data Count* section of the Developer Guide.

As root, download and install Counter Processor:

```
cd /usr/local
wget https://github.com/CDLUC3/counter-processor/archive/v0.0.1.tar.gz
tar xvfz v0.0.1.tar.gz
```

As root, change to the Counter Processor directory you just created, download the GeoLite2-Country tarball, untar it, and copy the geoup database into place:

```
cd /usr/local/counter-processor-0.0.1
wget https://geolite.maxmind.com/download/geoup/database/GeoLite2-Country.tar.gz
tar xvfz GeoLite2-Country.tar.gz
cp GeoLite2-Country_*/GeoLite2-Country.mmdb maxmind_geoup
```

As root, create a “counter” user and change ownership of Counter Processor directory to this new user:

```
useradd counter
chown -R counter:counter /usr/local/counter-processor-0.0.1
```

#### Installing Counter Processor Python Requirements

Counter Processor requires Python 3.6.4 or higher. The following commands are intended to be run as root but we are aware that Pythonistas might prefer fancy virtualenv or similar setups. Pull requests are welcome to improve these steps!

Enable the EPEL repo if you haven’t already:

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

Install Python 3.6:

```
yum install python36
```

Install Counter Processor Python requirements:

```
python3.6 -m ensurepip  
cd /usr/local/counter-processor-0.0.1  
pip3 install -r requirements.txt
```

See the *Make Data Count* section of the Admin Guide for how to configure and run Counter Processor.

### 4.3.10 Next Steps

Now that you have all the prerequisites in place, you can proceed to the *Installation* section.

## 4.4 Installation

Now that the *Prerequisites* are in place, we are ready to execute the Dataverse installation script (the “installer”) and verify that the installation was successful by logging in with a “superuser” account.

### Contents:

- *Running the Dataverse Installer*
- *Logging In*
  - *Superuser Account*
- *Troubleshooting*
  - *Dataset Cannot Be Published*
  - *Problems Sending Email*
  - *Mail Host Configuration & Authentication*
  - *UnknownHostException While Deploying*
- *Fresh Reinstall*
  - *Drop database*
  - *Clear Solr*
  - *Deleting Uploaded Files*
  - *Rerun Installer*

### 4.4.1 Running the Dataverse Installer

A scripted, interactive installer is provided. This script will configure your Glassfish environment, create the database, set some required options and start the application. Some configuration tasks will still be required after you run the installer! So make sure to consult the next section.

As mentioned in the *Prerequisites* section, RHEL/CentOS is the recommended Linux distribution. (The installer is also known to work on Mac OS X for setting up a development environment.)

Generally, the installer has a better chance of succeeding if you run it against a freshly installed Glassfish node that still has all the default configuration settings. In any event, please make sure that it is still configured to accept http connections on port 8080 - because that's where the installer expects to find the application once it's deployed.

You should have already downloaded the installer from <https://github.com/IQSS/dataverse/releases> when setting up and starting Solr under the *Prerequisites* section. Again, it's a zip file with "dvininstall" in the name.

Unpack the zip file - this will create the directory `dvininstall`.

**Important:** The installer will need to use the PostgreSQL command line utility `psql` in order to configure the database. If the executable is not in your system `PATH`, the installer will try to locate it on your system. However, we strongly recommend that you check and make sure it is in the `PATH`. This is especially important if you have multiple versions of PostgreSQL installed on your system. Make sure the `psql` that came with the version that you want to use with your Dataverse is the first on your path. For example, if the PostgreSQL distribution you are running is installed in `/Library/PostgreSQL/9.6`, add `/Library/PostgreSQL/9.6/bin` to the beginning of your `$PATH` variable. If you are *running* multiple PostgreSQL servers, make sure you know the port number of the one you want to use, as the installer will need it in order to connect to the database (the first PostgreSQL distribution installed on your system is likely using the default port 5432; but the second will likely be on 5433, etc.) Does every word in this paragraph make sense? If it does, great - because you definitely need to be comfortable with basic system tasks in order to install Dataverse. If not - if you don't know how to check where your PostgreSQL is installed, or what port it is running on, or what a `$PATH` is... it's not too late to stop. Because it will most likely not work. And if you contact us for help, these will be the questions we'll be asking you - so, again, you need to be able to answer them comfortably for it to work.

Execute the installer script like this (but first read the note below about not running the installer as root):

```
$ cd dvininstall
$ ./install
```

### It is no longer necessary to run the installer as root!

Just make sure the user running the installer has write permission to:

- `/usr/local/glassfish4/glassfish/lib`
- `/usr/local/glassfish4/glassfish/domains/domain1`
- the current working directory of the installer (it currently writes its logfile there), and
- your `jvm-option` specified `files.dir`

The only reason to run Glassfish as root would be to allow Glassfish itself to listen on the default HTTP(S) ports 80 and 443, or any other port below 1024. However, it is simpler and more secure to run Glassfish run on its default port of 8080 and hide it behind an Apache Proxy, via AJP, running on port 80 or 443. This configuration is required if you're going to use Shibboleth authentication. See more discussion on this here: *Shibboleth*.)

The script will prompt you for some configuration values. If this is a test/evaluation installation, it may be possible to accept the default values provided for most of the settings:

- Internet Address of your host: `localhost`
- Glassfish Directory: `/usr/local/glassfish4`
- Glassfish User: current user running the installer script
- Administrator email address for this Dataverse: `(none)`
- SMTP (mail) server to relay notification messages: `localhost`
- Postgres Server Address: `[127.0.0.1]`

- Postgres Server Port: 5432
- Postgres ADMIN password: secret
- Name of the Postgres Database: dvndb
- Name of the Postgres User: dvnapp
- Postgres user password: secret
- Remote Solr indexing service: LOCAL
- Rserve Server: localhost
- Rserve Server Port: 6311
- Rserve User Name: rserve
- Rserve User Password: rserve
- Administration Email address for the installation;
- Postgres admin password - We'll need it in order to create the database and user for the Dataverse to use, without having to run the installer as root. If you don't know your Postgres admin password, you may simply set the authorization level for localhost to "trust" in the PostgreSQL `pg_hba.conf` file (See the PostgreSQL section in the Prerequisites). If this is a production environment, you may want to change it back to something more secure, such as "password" or "md5", after the installation is complete.
- Network address of a remote Solr search engine service (if needed) - In most cases, you will be running your Solr server on the same host as the Dataverse application (then you will want to leave this set to the default value of LOCAL). But in a serious production environment you may set it up on a dedicated separate server.

If desired, these default values can be configured by creating a `default.config` (example here) file in the installer's working directory with new values (if this file isn't present, the above defaults will be used).

This allows the installer to be run in non-interactive mode (with `./install -y -f > install.out 2> install.err`), which can allow for easier interaction with automated provisioning tools.

All the Glassfish configuration tasks performed by the installer are isolated in the shell script `dvinstall/glassfish-setup.sh` (as `asadmin` commands).

**IMPORTANT:** As a security measure, the `glassfish-setup.sh` script stores passwords as "aliases" rather than plaintext. If you change your database password, for example, you will need to update the alias with `asadmin update-password-alias db_password_alias`, for example. Here is a list of the password aliases that are set by the installation process and entered into Glassfish's `domain.xml` file:

- `db_password_alias`
- `doi_password_alias`
- `rserve_password_alias`

Glassfish does not provide up to date documentation but Payara (a fork of Glassfish) does so for more information, please see <https://docs.payara.fish/documentation/payara-server/password-aliases/password-alias-asadmin-commands.html>

**IMPORTANT:** The installer will also ask for an external site URL for Dataverse. It is *imperative* that this value be supplied accurately, or a long list of functions will be inoperable, including:

- email confirmation links
- password reset links
- generating a Private URL
- exporting to Schema.org format (and showing JSON-LD in HTML's `<meta/>` tag)



- exporting to DDI format
- which Dataverse installation an “external tool” should return to
- which Dataverse installation Geoconnect should return to

**IMPORTANT:** Please note, that “out of the box” the installer will configure the Dataverse to leave unrestricted access to the administration APIs from (and only from) localhost. Please consider the security implications of this arrangement (anyone with shell access to the server can potentially mess with your Dataverse). An alternative solution would be to block open access to these sensitive API endpoints completely; and to only allow requests supplying a pre-defined “unblock token” (password). If you prefer that as a solution, please consult the supplied script `post-install-api-block.sh` for examples on how to set it up. See also “Securing Your Installation” under the *Configuration* section.

Dataverse uses *JHOVE* to help identify the file format (CSV, PNG, etc.) for files that users have uploaded. The installer places files called `jhove.conf` and `jhoveConfig.xsd` into the directory `/usr/local/glassfish4/glassfish/domains/domain1/config` by default and makes adjustments to the `jhove.conf` file based on the directory into which you chose to install Glassfish.

The script is to a large degree a derivative of the old installer from DVN 3.x. It is written in Perl. If someone in the community is eager to rewrite it, perhaps in a different language, please get in touch. :)

## 4.4.2 Logging In

Out of the box, Glassfish runs on port 8080 and 8181 rather than 80 and 443, respectively, so visiting <http://localhost:8080> (substituting your hostname) should bring up a login page. See the *Shibboleth* page for more on ports, but for now, let’s confirm we can log in by using port 8080. Poke a temporary hole in your firewall, if needed.

## Superuser Account

We’ll use the superuser account created by the installer to make sure you can log into Dataverse. For more on the difference between being a superuser and having the “Admin” role, read about configuring the root dataverse in the *Configuration* section.

(The `dvinstall/setup-all.sh` script, which is called by the installer sets the password for the superuser account and the username and email address come from a file it references at `dvinstall/data/user-admin.json`.)

Use the following credentials to log in:

- URL: <http://localhost:8080>
- username: `dataverseAdmin`
- password: `admin`

Congratulations! You have a working Dataverse installation. Soon you’ll be tweeting at [@dataverseorg](https://twitter.com/dataverseorg) asking to be added to the map at <http://dataverse.org> :)

Trouble? See if you find an answer in the troubleshooting section below.

Next you’ll want to check out the *Configuration* section, especially the section on security which reminds you to change the password above.

## 4.4.3 Troubleshooting

If the following doesn’t apply, please get in touch as explained in the *Introduction*. You may be asked to provide `glassfish4/glassfish/domains/domain1/logs/server.log` for debugging.

## Dataset Cannot Be Published

Check to make sure you used a fully qualified domain name when installing Dataverse. You can change the `dataverse.fqdn` JVM option after the fact per the *Configuration* section.

## Problems Sending Email

If your Dataverse installation is not sending system emails, you may need to provide authentication for your mail host. First, double check the SMTP server being used with this Glassfish `asadmin` command:

```
./asadmin get server.resources.mail-resource.mail/notifyMailSession.host
```

This should return the DNS of the mail host you configured during or after installation. `mail/notifyMailSession` is the JavaMail Session that's used to send emails to users.

If the command returns a host you don't want to use, you can modify your `notifyMailSession` with the Glassfish `asadmin set` command with necessary options ([click here for the manual page](#)), or via the admin console at <http://localhost:4848> with your domain running.

If your mail host requires a username/password for access, continue to the next section.

## Mail Host Configuration & Authentication

If you need to alter your mail host address, user, or provide a password to connect with, these settings are easily changed in the Glassfish admin console or via command line.

For the Glassfish console, load a browser with your domain online, navigate to <http://localhost:4848> and on the side panel find JavaMail Sessions. By default, Dataverse uses a session named `mail/notifyMailSession` for routing outgoing emails. Click this mail session in the window to modify it.

When fine tuning your JavaMail Session, there are a number of fields you can edit. The most important are:

- **Mail Host:** Desired mail host's DNS address (e.g. `smtp.gmail.com`)
- **Default User:** Username mail host will recognize (e.g. `user@gmail.com`)
- **Default Sender Address:** Email address that your Dataverse will send mail from

Depending on the SMTP server you're using, you may need to add additional properties at the bottom of the page (below "Advanced").

From the "Add Properties" utility at the bottom, use the "Add Property" button for each entry you need, and include the name / corresponding value as needed. Descriptions are optional, but can be used for your own organizational needs.

**Note:** These properties are just an example. You may need different/more/fewer properties all depending on the SMTP server you're using.

Name	Value
<code>mail.smtp.auth</code>	<code>true</code>
<code>mail.smtp.password</code>	[Default User password*]
<code>mail.smtp.port</code>	[Port number to route through]

**\*WARNING:** Entering a password here will *not* conceal it on-screen. It's recommended to use an *app password* (for `smtp.gmail.com` users) or utilize a dedicated/non-personal user account with SMTP server auths so that you do not risk compromising your password.

If your installation's mail host uses SSL (like `smtp.gmail.com`) you'll need these name/value pair properties in place:

Name	Value
mail.smtp.socketFactory.port	465
mail.smtp.port	465
mail.smtp.socketFactory.fallback	false
mail.smtp.socketFactory.class	javax.net.ssl.SSLSocketFactory

The mail session can also be set from command line. To use this method, you will need to delete your notifyMailSession and create a new one. See the below example:

- Delete: `./asadmin delete-javamail-resource mail/notifyMailSession`
- Create (remove brackets and replace the variables inside): `./asadmin create-javamail-resource --mailhost [smtp.gmail.com] --mailuser [test@test.com] --fromaddress [test@test.com] --property mail.smtp.auth=[true]:mail.smtp.password=[password]:mail.smtp.port=[465]:mail.smtp.socketFactory.port=[465]:mail.smtp.socketFactory.fallback=[false]:mail.smtp.socketFactory.class=[javax.net.ssl.SSLSocketFactory] mail/notifyMailSession`

Be sure you save the changes made here and then restart your Glassfish server to test it out.

### UnknownHostException While Deploying

If you are seeing “Caused by: java.net.UnknownHostException: myhost: Name or service not known” in server.log and your hostname is “myhost” the problem is likely that “myhost” doesn’t appear in `/etc/hosts`. See also <http://stackoverflow.com/questions/21817809/glassfish-exception-during-deployment-project-with-stateful-ejb/21850873#21850873>

## 4.4.4 Fresh Reinstall

Early on when you’re installing Dataverse, you may think, “I just want to blow away what I’ve installed and start over.” That’s fine. You don’t have to uninstall the various components like Glassfish, PostgreSQL and Solr, but you should be conscious of how to clear out their data.

### Drop database

In order to drop the database, you have to stop Glassfish, which will have open connections. Before you stop Glassfish, you may as well undeploy the war file. First, find the name like this:

```
./asadmin list-applications
```

Then undeploy it like this:

```
./asadmin undeploy dataverse-VERSION
```

Stop Glassfish with the init script provided in the *Prerequisites* section or just use:

```
./asadmin stop-domain
```

With Glassfish down, you should now be able to drop your database and recreate it:

```
psql -U dvnapp -c 'DROP DATABASE "dvndb"' template1
```

## Clear Solr

The database is fresh and new but Solr has stale data in it. Clear it out with this command:

```
curl http://localhost:8983/solr/collection1/update/json?commit=true
-H "Content-type: application/json" -X POST -d "{\"delete\": {
\"query\": \"*:*\"}\"}
```

## Deleting Uploaded Files

The path below will depend on the value for `dataverse.files.directory` as described in the *Configuration* section:

```
rm -rf /usr/local/glassfish4/glassfish/domains/domain1/files
```

## Rerun Installer

With all the data cleared out, you should be ready to rerun the installer per above.

Related to all this is a series of scripts at <https://github.com/IQSS/dataverse/blob/develop/scripts/deploy/phoenix.dataverse.org/deploy> that Dataverse developers use have the test server <http://phoenix.dataverse.org> rise from the ashes before integration tests are run against it. Your mileage may vary. :) For more on this topic, see “Rebuilding Your Dev Environment” in the *Development Environment* section of the Developer Guide.

## 4.5 Configuration

Now that you’ve successfully logged into Dataverse with a superuser account after going through a basic *Installation*, you’ll need to secure and configure your installation.

Settings within Dataverse itself are managed via JVM options or by manipulating values in the `setting` table directly or through API calls.

Once you have finished securing and configuring your Dataverse installation, you may proceed to the *Admin Guide* for more information on the ongoing administration of a Dataverse installation. Advanced configuration topics are covered in the *TwoRavens*, *Shibboleth* and *OAuth Login: ORCID, GitHub, Google* sections.

### Contents:

- *Securing Your Installation*
  - *Changing the Superuser Password*
  - *Blocking API Endpoints*
  - *Forcing HTTPS*
  - *Privacy Considerations*
    - \* *Email Privacy*
  - *Additional Recommendations*
  - *Run Glassfish as a User Other Than Root*
  - *Enforce Strong Passwords for User Accounts*
- *Network Ports*

- *Root Dataverse Permissions*
- *Persistent Identifiers and Publishing Datasets*
  - *Configuring Dataverse for DOIs*
  - *Configuring Dataverse for Handles*
- *Auth Modes: Local vs. Remote vs. Both*
  - *Local Only Auth*
  - *Both Local and Remote Auth*
  - *Remote Only Auth*
- *File Storage: Local Filesystem vs. Swift vs. S3*
  - *Swift Storage*
  - *Setting up Compute*
  - *Amazon S3 Storage (or Compatible)*
    - \* *First: Set Up Accounts and Access Credentials*
      - *Preparation When Using Amazon's S3 Service*
      - *Preparation When Using Custom S3-Compatible Service*
      - *Reported Working S3-Compatible Storage*
      - *Manually Set Up Credentials File*
      - *Console Commands to Set Up Access Configuration*
    - \* *Second: Configure Dataverse to use S3 Storage*
      - *S3 Storage Options*
- *Branding Your Installation*
  - *Custom Homepage*
  - *Custom Navbar Logo*
  - *Custom Header*
  - *Custom Footer*
  - *Custom Stylesheet*
- *Internationalization*
  - *Adding Multiple Languages to the Dropdown in the Header*
  - *Configuring the "lang" Directory*
  - *Creating a languages.zip File*
  - *Load the languages.zip file into Dataverse*
  - *How to Help Translate Dataverse Into Your Language*
- *Web Analytics Code*
  - *Tracking Button Clicks*
- *BagIt Export*

- *Duracloud Configuration*
- *Local Path Configuration*
- *API Call*
- *PostPublication Workflow*
- *Going Live: Launching Your Production Deployment*
  - *Letting Search Engines Crawl Your Installation*
    - \* *Ensure robots.txt Is Not Blocking Search Engines*
    - \* *Creating a Sitemap and Submitting it to Search Engines*
  - *Putting Your Dataverse Installation on the Map at dataverse.org*
  - *Administration of Your Dataverse Installation*
  - *Setting Up Integrations*
- *JVM Options*
  - *dataverse.fqdn*
  - *dataverse.siteUrl*
  - *dataverse.files.directory*
  - *dataverse.auth.password-reset-timeout-in-minutes*
  - *dataverse.rserve.host*
  - *dataverse.rserve.port*
  - *dataverse.rserve.user*
  - *dataverse.rserve.tempdir*
  - *dataverse.rserve.password*
  - *dataverse.dropbox.key*
  - *dataverse.path.imagemagick.convert*
  - *dataverse.dataAccess.thumbnail.image.limit*
  - *dataverse.dataAccess.thumbnail.pdf.limit*
  - *doi.baseurlstring*
  - *doi.username*
  - *doi.password*
  - *dataverse.handlenet.admcredfile*
  - *dataverse.handlenet.admprivphrase*
  - *dataverse.handlenet.index*
  - *dataverse.timerServer*
  - *dataverse.lang.directory*
  - *dataverse.files.hide-schema-dot-org-download-urls*
- *Database Settings*

- *:BlockedApiPolicy*
- *:BlockedApiEndpoints*
- *:BlockedApiKey*
- *BuiltinUsers.KEY*
- *:SearchApiRequiresToken*
- *:SystemEmail*
- *:HomePageCustomizationFile*
- *:LogoCustomizationFile*
- *:HeaderCustomizationFile*
- *:DisableRootDataverseTheme*
- *:FooterCustomizationFile*
- *:StyleCustomizationFile*
- *:WebAnalyticsCode*
- *:FooterCopyright*
- *:DoiProvider*
- *:Protocol*
- *:Authority*
- *:Shoulder*
- *:IdentifierGenerationStyle*
- *:DataFilePIDFormat*
- *:FilePIDsEnabled*
- *:IndependentHandleService*
- *:ApplicationTermsOfUse*
- *:ApplicationPrivacyPolicyUrl*
- *:ApiTermsOfUse*
- *:ExcludeEmailFromExport*
- *:NavbarAboutUrl*
- *:GuidesBaseUrl*
- *:GuidesVersion*
- *:NavbarSupportUrl*
- *:MetricsUrl*
- *:StatusMessageHeader*
- *:StatusMessageText*
- *:MaxFileUploadSizeInBytes*
- *:ZipDownloadLimit*

- *:TabularIngestSizeLimit*
- *:ZipUploadFilesLimit*
- *:SolrHostColonPort*
- *:SolrFullTextIndexing*
- *:SolrMaxFileSizeForFullTextIndexing*
- *:SignUpUrl*
- *:LoginSessionTimeout*
- *:TwoRavensUrl*
- *:TwoRavensTabularView*
- *:GeoconnectCreateEditMaps*
- *:GeoconnectViewMaps*
- *:DatasetPublishPopupCustomText*
- *:DatasetPublishPopupCustomTextOnAllVersions*
- *:SearchHighlightFragmentSize*
- *:ScrubMigrationData*
- *:MinutesUntilConfirmEmailTokenExpires*
- *:DefaultAuthProvider*
- *:AllowSignUp*
- *:FileFixityChecksumAlgorithm*
- *:PVMinLength*
- *:PVMaxLength*
- *:PVNumberOfConsecutiveDigitsAllowed*
- *:PVCharacterRules*
- *:PVNumberOfCharacteristics*
- *:PVDictionaries*
- *:PVGoodStrength*
- *:PVCustomPasswordResetAlertMessage*
- *:ShibPassiveLoginEnabled*
- *:ComputeBaseUrl*
- *:CloudEnvironmentName*
- *:PublicInstall*
- *:DataCaptureModuleUrl*
- *:RepositoryStorageAbstractionLayerUrl*
- *:UploadMethods*
- *:DownloadMethods*



- `:GuestbookResponsesPageDisplayLimit`
- `:CustomDatasetSummaryFields`
- `:AllowApiTokenLookupViaApi`
- `:ProvCollectionEnabled`
- `:MetricsCacheTimeoutMinutes`
- `:MDCLogPath`
- `:Languages`
- `:InheritParentRoleAssignments`
- `:AllowCors`

## 4.5.1 Securing Your Installation

### Changing the Superuser Password

The default password for the “dataverseAdmin” superuser account is “admin”, as mentioned in the *Installation* section, and you should change it, of course.

### Blocking API Endpoints

The *Native API* contains a useful but potentially dangerous API endpoint called “admin” that allows you to change system settings, make ordinary users into superusers, and more. The `builtin-users` endpoint lets people create a local/builtin user account if they know the `BuiltinUsers.KEY` value described below.

By default, all APIs can be operated on remotely and a number of endpoints do not require authentication. <https://github.com/IQSS/dataverse/issues/1886> was opened to explore changing these defaults, but until then it is very important to block both the “admin” endpoint (and at least consider blocking `builtin-users`). For details please see also the section on `:BlockedApiPolicy` below.

It’s also possible to prevent file uploads via API by adjusting the `:UploadMethods` database setting.

### Forcing HTTPS

To avoid having your users send credentials in the clear, it’s strongly recommended to force all web traffic to go through HTTPS (port 443) rather than HTTP (port 80). The ease with which one can install a valid SSL cert into Apache compared with the same operation in Glassfish might be a compelling enough reason to front Glassfish with Apache. In addition, Apache can be configured to rewrite HTTP to HTTPS with rules such as those found at <https://wiki.apache.org/httpd/RewriteHTTPToHTTPS> or in the section on *Shibboleth*.

## Privacy Considerations

### Email Privacy

Out of the box, Dataverse will list email addresses of the contacts for datasets when users visit a dataset page and click the “Export Metadata” button. Additionally, out of the box, Dataverse will list email addresses of dataverse contacts via API (see *View a Dataverse* in the *Native API* section of the API Guide). If you would like to exclude these email addresses from export, set `:ExcludeEmailFromExport` to true.

## Additional Recommendations

### Run Glassfish as a User Other Than Root

See the Glassfish section of *Prerequisites* for details and init scripts for running Glassfish as non-root.

Related to this is that you should remove `/root/.glassfish/pass` to ensure that Glassfish isn't ever accidentally started as root. Without the password, Glassfish won't be able to start as root, which is a good thing.

### Enforce Strong Passwords for User Accounts

Dataverse only stores passwords (as salted hash, and using a strong hashing algorithm) for “builtin” users. You can increase the password complexity rules to meet your security needs. If you have configured your Dataverse installation to allow login from remote authentication providers such as Shibboleth, ORCID, GitHub or Google, you do not have any control over those remote providers' password complexity rules. See the “Auth Modes: Local vs. Remote vs. Both” section below for more on login options.

Even if you are satisfied with the out-of-the-box password complexity rules Dataverse ships with, for the “dataverseAdmin” account you should use a strong password so the hash cannot easily be cracked through dictionary attacks.

Password complexity rules for “builtin” accounts can be adjusted with a variety of settings documented below. Here's a list:

- `:PVMinLength`
- `:PVMaxLength`
- `:PVNumberOfConsecutiveDigitsAllowed`
- `:PVCharacterRules`
- `:PVNumberOfCharacteristics`
- `:PVDictionaries`
- `:PVGoodStrength`
- `:PVCustomPasswordResetAlertMessage`

## 4.5.2 Network Ports

Remember how under “Decisions to Make” in the *Preparation* section we mentioned you'll need to make a decision about whether or not to introduce a proxy in front of Dataverse such as Apache or nginx? The time has come to make that decision.

The need to redirect port HTTP (port 80) to HTTPS (port 443) for security has already been mentioned above and the fact that Glassfish puts these services on 8080 and 8181, respectively, was touched on in the *Installation* section. In production, you don't want to tell your users to use Dataverse on ports 8080 and 8181. You should have them use the standard HTTPS port, which is 443.

Your decision to proxy or not should primarily be driven by which features of Dataverse you'd like to use. If you'd like to use Shibboleth, the decision is easy because proxying or “fronting” Glassfish with Apache is required. The details are covered in the *Shibboleth* section.

If you'd like to use TwoRavens, you should also consider fronting with Apache because you will be required to install an Apache anyway to make use of the rApache module. For details, see the *TwoRavens* section.

Even if you have no interest in Shibboleth nor TwoRavens, you may want to front Dataverse with Apache or nginx to simply the process of installing SSL certificates. There are many tutorials on the Internet for adding certs to Apache, including a some [notes used by the Dataverse team](#), but the process of adding a certificate to Glassfish is arduous and

not for the faint of heart. The Dataverse team cannot provide much help with adding certificates to Glassfish beyond linking to [tips](#) on the web.

Still not convinced you should put Glassfish behind another web server? Even if you manage to get your SSL certificate into Glassfish, how are you going to run Glassfish on low ports such as 80 and 443? Are you going to run Glassfish as root? Bad idea. This is a security risk. Under “Additional Recommendations” under “Securing Your Installation” above you are advised to configure Glassfish to run as a user other than root. (The Dataverse team will close <https://github.com/IQSS/dataverse/issues/1934> after updating the Glassfish init script provided in the *Prerequisites* section to not require root.)

There’s also the issue of serving a production-ready version of robots.txt. By using a proxy such as Apache, this is a one-time “set it and forget it” step as explained below in the “Going Live” section.

If you are convinced you’d like to try fronting Glassfish with Apache, the *Shibboleth* section should be good resource for you.

If you really don’t want to front Glassfish with any proxy (not recommended), you can configure Glassfish to run HTTPS on port 443 like this:

```
./asadmin set server-config.network-config.network-listeners.network-listener.  
http-listener-2.port=443
```

What about port 80? Even if you don’t front Dataverse with Apache, you may want to let Apache run on port 80 just to rewrite HTTP to HTTPS as described above. You can use a similar command as above to change the HTTP port that Glassfish uses from 8080 to 80 (substitute `http-listener-1.port=80`). Glassfish can be used to enforce HTTPS on its own without Apache, but configuring this is an exercise for the reader. Answers here may be helpful: <http://stackoverflow.com/questions/25122025/glassfish-v4-java-7-port-unification-error-not-able-to-redirect-http-to>

If you are running an installation with Apache and Glassfish on the same server, and would like to restrict Glassfish from responding to any requests to port 8080 from external hosts (in other words, not through Apache), you can restrict the AJP listener to localhost only with:

```
./asadmin set server-config.network-config.network-listeners.network-listener.  
http-listener-1.address=127.0.0.1
```

You should **NOT** use the configuration option above if you are running in a load-balanced environment, or otherwise have the web server on a different host than the application server.

### 4.5.3 Root Dataverse Permissions

The user who creates a dataverse is given the “Admin” role on that dataverse. The root dataverse is created automatically for you by the installer and the “Admin” is the superuser account (“dataverseAdmin”) we used in the *Installation* section to confirm that we can log in. These next steps of configuring the root dataverse require the “Admin” role on the root dataverse, but not the much more powerful superuser attribute. In short, users with the “Admin” role are subject to the permission system. A superuser, on the other hand, completely bypasses the permission system. You can give non-superusers the “Admin” role on the root dataverse if you’d like them to configure the root dataverse.

In order for non-superusers to start creating dataverses or datasets, you need click “Edit” then “Permissions” and make choices about which users can add dataverses or datasets within the root dataverse. (There is an API endpoint for this operation as well.) Again, the user who creates a dataverse will be granted the “Admin” role on that dataverse. Non-superusers who are not “Admin” on the root dataverse will not be able to do anything useful until the root dataverse has been published.

As the person installing Dataverse you may or may not be a local metadata expert. You may want to have others sign up for accounts and grant them the “Admin” role at the root dataverse to configure metadata fields, templates, browse/search facets, guestbooks, etc. For more on these topics, consult the *Dataverse Management* section of the User Guide.

## 4.5.4 Persistent Identifiers and Publishing Datasets

Persistent identifiers are a required and integral part of the Dataverse platform. They provide a URL that is guaranteed to resolve to the datasets or files they represent. Dataverse currently supports creating identifiers using DOI and Handle.

By default, the installer configures a default DOI namespace (10.5072) with DataCite as the registration provider. Please note that as of the release 4.9.3, we can no longer use EZID as the provider. Unlike EZID, DataCite requires that you register for a test account, configured with your own prefix (please contact [support@datacite.org](mailto:support@datacite.org)). Once you receive the login name, password, and prefix for the account, configure the credentials in your `domain.xml`, as the following two JVM options:

```
<jvm-options>-Ddoi.username=...</jvm-options>
<jvm-options>-Ddoi.password=...</jvm-options>
```

and restart Glassfish. The prefix can be configured via the API (where it is referred to as “Authority”):

```
curl -X PUT -d 10.xxxx http://localhost:8080/api/admin/settings/:Authority
```

Once this is done, you will be able to publish datasets and files, but the persistent identifiers will not be citable, and they will only resolve from the DataCite test environment (and then only if the Dataverse from which you published them is accessible - DOIs minted from your laptop will not resolve). Note that any datasets or files created using the test configuration cannot be directly migrated and would need to be created again once a valid DOI namespace is configured.

To properly configure persistent identifiers for a production installation, an account and associated namespace must be acquired for a fee from a DOI or HDL provider. **DataCite** (<https://www.datacite.org>) is the recommended DOI provider (see <https://dataverse.org/global-dataverse-community-consortium> for more on joining DataCite) but **EZID** (<http://ezid.cdlib.org>) is an option for the University of California according to <https://www.cdlib.org/cdlinfo/2017/08/04/ezid-doi-service-is-evolving/>. **Handle.Net** (<https://www.handle.net>) is the HDL provider.

Once you have your DOI or Handle account credentials and a namespace, configure Dataverse to use them using the JVM options and database settings below.

### Configuring Dataverse for DOIs

By default Dataverse attempts to register DOIs for each dataset and file under a test authority, though you must apply for your own credentials as explained above.

Here are the configuration options for DOIs:

#### JVM Options:

- `doi.baseurlstring`
- `doi.username`
- `doi.password`

#### Database Settings:

- `:DoiProvider`
- `:Protocol`
- `:Authority`
- `:Shoulder`
- `:IdentifierGenerationStrategy` (optional)
- `:DataFilePIDFormat` (optional)

- `:FilePIDsEnabled` (optional, defaults to true)

## Configuring Dataverse for Handles

Here are the configuration options for handles:

### JVM Options:

- `dataverse.handlenet.admcredfile`
- `dataverse.handlenet.admprivphrase`
- `dataverse.handlenet.index`

### Database Settings:

- `:Protocol`
- `:Authority`
- `:IdentifierGenerationStyle` (optional)
- `:DataFilePIDFormat` (optional)
- `:IndependentHandleService` (optional)

Note: If you are **minting your own handles** and plan to set up your own handle service, please refer to [Handle.Net](#) documentation.

## 4.5.5 Auth Modes: Local vs. Remote vs. Both

There are three valid configurations or modes for authenticating users to Dataverse:

### Local Only Auth

Out of the box, Dataverse is configured in “local only” mode. The “dataverseAdmin” superuser account mentioned in the *Installation* section is an example of a local account. Internally, these accounts are called “builtin” because they are built in to the Dataverse application itself.

### Both Local and Remote Auth

The `authenticationprovider` database table controls which “authentication providers” are available within Dataverse. Out of the box, a single row with an id of “builtin” will be present. For each user in Dataverse, the `authenticateduserlookup` table will have a value under `authenticationproviderid` that matches this id. For example, the default “dataverseAdmin” user will have the value “builtin” under `authenticationproviderid`. Why is this important? Users are tied to a specific authentication provider but conversion mechanisms are available to switch a user from one authentication provider to the other. As explained in the *Account Creation + Management* section of the User Guide, a graphical workflow is provided for end users to convert from the “builtin” authentication provider to a remote provider. Conversion from a remote authentication provider to the builtin provider can be performed by a sysadmin with access to the “admin” API. See the *Native API* section of the API Guide for how to list users and authentication providers as JSON.

Adding and enabling a second authentication provider (*Add Authentication Provider* and *Enable or Disable an Authentication Provider*) will result in the Log In page showing additional providers for your users to choose from. By default, the Log In page will show the “builtin” provider, but you can adjust this via the `:DefaultAuthProvider` configuration option. Further customization can be achieved by setting `:AllowSignUp` to “false”, thus preventing users from creating local accounts via the web interface. Please note that local accounts can also be created via API, and the

way to prevent this is to block the `builtin-users` endpoint (*:BlockedApiEndpoints*) or scramble (or remove) the `BuiltinUsers.KEY` database setting (*BuiltinUsers.KEY*) per the *Configuration* section.

To configure Shibboleth see the *Shibboleth* section and to configure OAuth see the *OAuth Login: ORCID, GitHub, Google* section.

### Remote Only Auth

As for the “Remote only” authentication mode, it means that:

- Shibboleth or OAuth has been enabled.
- `:AllowSignUp` is set to “false” to prevent users from creating local accounts via the web interface.
- `:DefaultAuthProvider` has been set to use the desired authentication provider
- The “builtin” authentication provider has been disabled (*Enable or Disable an Authentication Provider*). Note that disabling the “builtin” authentication provider means that the API endpoint for converting an account from a remote auth provider will not work. Converting directly from one remote authentication provider to another (i.e. from GitHub to Google) is not supported. Conversion from remote is always to “builtin”. Then the user initiates a conversion from “builtin” to remote. Note that longer term, the plan is to permit multiple login options to the same Dataverse account per <https://github.com/IQSS/dataverse/issues/3487> (so all this talk of conversion will be moot) but for now users can only use a single login option, as explained in the *Account Creation + Management* section of the User Guide. In short, “remote only” might work for you if you only plan to use a single remote authentication provider such that no conversion between remote authentication providers will be necessary.

### 4.5.6 File Storage: Local Filesystem vs. Swift vs. S3

By default, a Dataverse installation stores data files (files uploaded by end users) on the filesystem at `/usr/local/glassfish4/glassfish/domains/domain1/files` but this path can vary based on answers you gave to the installer (see the *Running the Dataverse Installer* section of the Installation Guide) or afterward by reconfiguring the `dataverse.files.directory` JVM option described below.

#### Swift Storage

Rather than storing data files on the filesystem, you can opt for an experimental setup with a *Swift Object Storage* backend. Each dataset that users create gets a corresponding “container” on the Swift side, and each data file is saved as a file within that container.

**In order to configure a Swift installation**, you need to complete these steps to properly modify the JVM options:

First, run all the following create commands with your Swift endpoint information and credentials:

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.  
↪defaultEndpoint=endpoint1"  
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.authType.  
↪endpoint1=your-auth-type"  
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.authUrl.  
↪endpoint1=your-auth-url"  
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.tenant.  
↪endpoint1=your-tenant-name"  
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.username.  
↪endpoint1=your-username"  
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.endpoint.  
↪endpoint1=your-swift-endpoint"
```

`auth_type` can either be `keystone`, `keystone_v3`, or it will assumed to be `basic`. `auth_url` should be your keystone authentication URL which includes the tokens (e.g. for `keystone`, `https://openstack.example.edu:35357/v2.0/tokens` and for `keystone_v3`, `https://openstack.example.edu:35357/v3/auth/tokens`). `swift_endpoint` is a URL that looks something like `http://rdgw.swift.example.org/swift/v1`.

Then create a password alias by running (without changes):

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.password.  
↪endpoint1='\${ALIAS=swiftpassword-alias}' "  
./asadmin $ASADMIN_OPTS create-password-alias swiftpassword-alias
```

The second command will trigger an interactive prompt asking you to input your Swift password.

Second, update the JVM option `dataverse.files.storage-driver-id` by running the delete command:

```
./asadmin $ASADMIN_OPTS delete-jvm-options "\-Ddataverse.files.  
storage-driver-id=file"
```

Then run the create command:

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.  
storage-driver-id=swift"
```

You also have the option to set a **custom container name separator**. It is initialized to `_`, but you can change it by running the create command:

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.  
folderPathSeparator=-"
```

By default, your Swift installation will be `public-only`, meaning users will be unable to put access restrictions on their data. If you are comfortable with this level of privacy, the final step in your setup is to set the `:PublicInstall` setting to `true`.

In order to **enable file access restrictions**, you must enable Swift to use temporary URLs for file access. To enable usage of temporary URLs, set a hash key both on your swift endpoint and in your `swift.properties` file. You can do so by running the create command:

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.hashKey.  
endpoint1=your-hash-key"
```

You also have the option to set a custom expiration length, in seconds, for a generated temporary URL. It is initialized to 60 seconds, but you can change it by running the create command:

```
./asadmin $ASADMIN_OPTS create-jvm-options "\-Ddataverse.files.swift.  
temporaryUrlExpiryTime=3600"
```

In this example, you would be setting the expiration length for one hour.

## Setting up Compute

Once you have configured a Swift Object Storage backend, you also have the option of enabling a connection to a computing environment. To do so, you need to configure the database settings for `:ComputeBaseUrl` and `:CloudEnvironmentName`.

Once you have set up `:ComputeBaseUrl` properly in both Dataverse and your cloud environment, validated users will have three options for accessing the computing environment:

- Compute on a single dataset
- Compute on multiple datasets

- Compute on a single datafile

The compute buttons on dataset and file pages will link validated users to your computing environment. If a user is computing on one dataset, the compute button will redirect to:

```
:ComputeBaseUrl?datasetPersistentId
```

If a user is computing on multiple datasets, the compute button will redirect to:

```
:ComputeBaseUrl/multiparty?datasetPersistentId&anotherDatasetPersistentId&anotherDatasetPe  
..
```

If a user is computing on a single file, depending on the configuration of your installation, the compute button will either redirect to:

```
:ComputeBaseUrl?datasetPersistentId=yourObject
```

if your installation's `:PublicInstall` setting is true, or:

```
:ComputeBaseUrl?datasetPersistentId=yourObject&temp_url_sig=yourTempUrlSig&temp_url_expire
```

You can configure this redirect properly in your cloud environment to generate a temporary URL for access to the Swift objects for computing.

### Amazon S3 Storage (or Compatible)

For institutions and organizations looking to use some kind of S3-based object storage for files uploaded to Dataverse, this is entirely possible. You can either use Amazon Web Services or use some other, even on-site S3-compatible storage (like Minio, Ceph RADOS S3 Gateway and many more).

**Note:** The Dataverse Team is most familiar with AWS S3, and can provide support on its usage with Dataverse. Thanks to community contributions, the application's architecture also allows non-AWS S3 providers. The Dataverse Team can provide very limited support on these other providers. We recommend reaching out to the wider Dataverse community if you have questions.

### First: Set Up Accounts and Access Credentials

Dataverse and the AWS SDK make use of the “AWS credentials profile file” and “AWS config profile file” located in `~/ .aws/` where `~` is the home directory of the user you run Glassfish as. This file can be generated via either of two methods described below:

1. Manually through creation of the credentials and config files or
2. Automatically via the AWS console commands.

### Preparation When Using Amazon's S3 Service

You'll need an AWS account with an associated S3 bucket for your installation to use. From the S3 management console (e.g. <https://console.aws.amazon.com/>), you can poke around and get familiar with your bucket.

**Make note** of the **bucket's name** and the **region** its data is hosted in.

To **create a user** with full S3 access and nothing more for security reasons, we recommend using IAM (Identity and Access Management). See [IAM User Guide](#) for more info on this process.

**Generate the user keys** needed for Dataverse afterwards by clicking on the created user. (You can skip this step when running on EC2, see below.)



---

**Tip:** If you are hosting Dataverse on an AWS EC2 instance alongside storage in S3, it is possible to use IAM Roles instead of the credentials file (the file at `~/.aws/credentials` mentioned below). Please note that you will still need the `~/.aws/config` file to specify the region. For more information on this option, see <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html>

---

## Preparation When Using Custom S3-Compatible Service

We assume you have your S3-compatible custom storage in place, up and running, ready for service.

Please make note of the following details:

- **Endpoint URL** - consult the documentation of your service on how to find it.
  - Example: `https://play.minio.io:9000`
- **Region:** Optional, but some services might use it. Consult your service documentation.
  - Example: `us-east-1`
- **Access key ID and secret access key:** Usually you can generate access keys within the user profile of your service.
  - Example:
    - \* ID: `Q3AM3UQ867SPQQA43P2F`
    - \* Key: `zuf+tfteSlswRu7BJ86wekitnifILbZam1KYY3TG`
- **Bucket name:** Dataverse will fail opening and uploading files on S3 if you don't create one.
  - Example: `dataverse`

## Reported Working S3-Compatible Storage

**Minio v2018-09-12** Set `dataverse.files.s3-path-style-access=true`, as Minio works path-based. Works pretty smooth, easy to setup. **Can be used for quick testing, too:** just use the example values above. Uses the public (read: unsecure and possibly slow) <https://play.minio.io:9000> service.

**HINT:** If you are successfully using an S3 storage implementation not yet listed above, please feel free to [open an issue at Github](#) and describe your setup. We will be glad to add it here.

## Manually Set Up Credentials File

To create the `~/.aws/credentials` file manually, you will need to generate a key/secret key (see above). Once you have acquired the keys, they need to be added to the `credentials` file. The format for credentials is as follows:

```
[default]
aws_access_key_id = <insert key, no brackets>
aws_secret_access_key = <insert secret key, no brackets>
```

While using Amazon's service, you must also specify the AWS region in the `~/.aws/config` file, for example:

```
[default]
region = us-east-1
```

Place these two files in a folder named `.aws` under the home directory for the user running your Dataverse Glassfish instance. (From the [AWS Command Line Interface Documentation](#): “In order to separate credentials from less sensitive options, region and output format are stored in a separate file named `config` in the same folder”)

## Console Commands to Set Up Access Configuration

Begin by installing the CLI tool `pip` to install the [AWS command line interface](#) if you don't have it.

First, we'll get our access keys set up. If you already have your access keys configured, skip this step. From the command line, run:

- `pip install awscli`
- `aws configure`

You'll be prompted to enter your Access Key ID and secret key, which should be issued to your AWS account. The subsequent config steps after the access keys are up to you. For reference, the keys will be stored in `~/.aws/credentials`, and your AWS access region in `~/.aws/config`.

**TIP: When using a custom S3 URL endpoint, you need to add it to every `aws` call: `aws --endpoint-url <URL> s3 ...`** (you may omit it while configuring).

## Second: Configure Dataverse to use S3 Storage

With access to your bucket in place, we'll want to navigate to `/usr/local/glassfish4/glassfish/bin/` and execute the following `asadmin` commands to set up the proper JVM options. Recall that out of the box, Dataverse is configured to use local file storage. You'll need to delete the existing storage driver before setting the new one.

```
./asadmin $ASADMIN_OPTS delete-jvm-options "-Ddataverse.files.storage-driver-id=file"
./asadmin $ASADMIN_OPTS create-jvm-options "-Ddataverse.files.storage-driver-id=s3"
```

Then, we'll need to identify which S3 bucket we're using. Replace `your_bucket_name` with, of course, your bucket:

```
./asadmin create-jvm-options "-Ddataverse.files.s3-bucket-name=your_bucket_name"
```

Optionally, you can have users download files from S3 directly rather than having files pass from S3 through Glassfish to your users. To accomplish this, set `dataverse.files.s3-download-redirect` to `true` like this:

```
./asadmin create-jvm-options "-Ddataverse.files.s3-download-redirect=true"
```

If you enable `dataverse.files.s3-download-redirect` as described above, note that the S3 URLs expire after an hour by default but you can configure the expiration time using the `dataverse.files.s3-url-expiration-minutes` JVM option. Here's an example of setting the expiration time to 120 minutes:

```
./asadmin create-jvm-options "-Ddataverse.files.s3-url-expiration-minutes=120"
```

In case you would like to configure Dataverse to use a custom S3 service instead of Amazon S3 services, please add the options for the custom URL and region as documented below. Please read above if your desired combination has been tested already and what other options have been set for a successful integration.

Lastly, go ahead and restart your glassfish server. With Dataverse deployed and the site online, you should be able to upload datasets and data files and see the corresponding files in your S3 bucket. Within a bucket, the folder structure emulates that found in local file storage.

## S3 Storage Options

JVM Option	Value	Description	Default value
<code>dataverse.files.storage-driver-id</code>	<code>s3</code>	Enable S3 storage driver.	<code>file</code>
<code>dataverse.files.s3-bucket-name</code>	<code>&lt;?&gt;</code>	The bucket name. See above.	<code>(none)</code>
<code>dataverse.files.s3-download-redirect</code>	<code>true/false</code>	Enable direct download or proxy through Dataverse.	<code>false</code>
<code>dataverse.files.s3-url-expiration-minutes</code>	<code>&lt;?&gt;</code>	If direct downloads: time until links expire. Optional.	<code>60</code>
<code>dataverse.files.s3-custom-endpoint-url</code>	<code>&lt;?&gt;</code>	Use custom S3 endpoint. Needs URL either with or without protocol.	<code>(none)</code>
<code>dataverse.files.s3-custom-endpoint-region</code>	<code>&lt;?&gt;</code>	Only used when using custom endpoint. Optional.	<code>dataverse</code>
<code>dataverse.files.s3-path-style-access</code>	<code>true/false</code>	Use path style buckets instead of subdomains. Optional.	<code>false</code>

## 4.5.7 Branding Your Installation

The name of your root dataverse is the brand of your installation of Dataverse and appears in various places such as notifications and support links, as outlined in the *:SystemEmail* section below. To further brand your installation and make it your own, Dataverse provides configurable options for easy-to-add (and maintain) custom branding for your Dataverse installation. Here are the custom branding and content options you can add:

- Custom welcome/homepage
- Logo image to navbar
- Header
- Footer
- CSS stylesheet

Downloadable sample HTML and CSS files are provided below which you can edit as you see fit. It's up to you to create a directory in which to store these files, such as `/var/www/dataverse` in the examples below.

You may also want to look at samples at <https://github.com/shlake/LibraDataHomepage> from community member Sherry Lake as well as her poster from the Dataverse Community Meeting 2018 called "Branding Your Local Dataverse": <https://github.com/IQSS/dataverse/files/2128735/UVaHomePage.pdf>

A simpler option to brand and customize your installation is to utilize the dataverse theme, which each dataverse has, that allows you to change colors, add a logo, tagline or website link to the dataverse header section of the page. Those options are outlined in the *Dataverse Management* section of the User Guide.

### Custom Homepage

Dataverse allows you to use a custom homepage or welcome page in place of the default root dataverse page. This allows for complete control over the look and feel of your installation's homepage.

Download this sample: `custom-homepage.html` and place it at `/var/www/dataverse/branding/custom-homepage.html`.

Once you have the location of your custom homepage HTML file, run this curl command to add it to your settings:

```
curl -X PUT -d '/var/www/dataverse/branding/custom-homepage.html' http://localhost:8080/api/admin/settings/:HomePageCustomizationFile
```

If you prefer to start with less of a blank slate, you can download the `custom-homepage-dynamic.html` template which was built for the Harvard Dataverse, and includes branding messaging, action buttons, search input, subject links, and recent dataset links. This page was built to utilize the *Metrics API* to deliver dynamic content to the page via javascript.

Note that the `custom-homepage.html` and `custom-homepage-dynamic.html` files provided have multiple elements that assume your root dataverse still has an alias of “root”. While you were branding your root dataverse, you may have changed the alias to “harvard” or “librascholar” or whatever and you should adjust the custom homepage code as needed.

For more background on what this curl command above is doing, see the “Database Settings” section below. If you decide you’d like to remove this setting, use the following curl command:

```
curl -X DELETE http://localhost:8080/api/admin/settings/:HomePageCustomizationFile
```

### Custom Navbar Logo

Dataverse allows you to replace the default Dataverse icon and name branding in the navbar with your own custom logo. Note that this logo is separate from the *root dataverse theme* logo.

The custom logo image file is expected to be small enough to fit comfortably in the navbar, no more than 50 pixels in height and 160 pixels in width. Create a `navbar` directory in your Glassfish `logos` directory and place your custom logo there. By Glassfish default, your logo image file will be located at `/usr/local/glassfish4/glassfish/domains/domain1/docroot/logos/navbar/logo.png`.

Once you have the location of your custom logo image file, run this curl command to add it to your settings:

```
curl -X PUT -d '/logos/navbar/logo.png' http://localhost:8080/api/admin/settings/:LogoCustomizationFile
```

### Custom Header

Download this sample: `custom-header.html` and place it at `/var/www/dataverse/branding/custom-header.html`.

Once you have the location of your custom header HTML file, run this curl command to add it to your settings:

```
curl -X PUT -d '/var/www/dataverse/branding/custom-header.html' http://localhost:8080/api/admin/settings/:HeaderCustomizationFile
```

If you have enabled a custom header or navbar logo, you might prefer to disable the theme of the root dataverse. You can do so by setting `:DisableRootDataverseTheme` to `true` like this:

```
curl -X PUT -d 'true' http://localhost:8080/api/admin/settings/:DisableRootDataverseTheme
```

Please note: Disabling the display of the root dataverse theme also disables your ability to edit it. Remember that dataverse owners can set their dataverses to “inherit theme” from the root. Those dataverses will continue to inherit the root dataverse theme (even though it no longer displays on the root). If you would like to edit the root dataverse theme in the future, you will have to re-enable it first.

### Custom Footer

Download this sample: `custom-footer.html` and place it at `/var/www/dataverse/branding/custom-footer.html`.

Once you have the location of your custom footer HTML file, run this curl command to add it to your settings:

```
curl -X PUT -d '/var/www/dataverse/branding/custom-footer.html' http://localhost:8080/api/admin/settings/:FooterCustomizationFile
```

### Custom Stylesheet

You can style your custom homepage, footer and header content with a custom CSS file. With advanced CSS know-how, you can achieve custom branding and page layouts by utilizing `position`, `padding` or `margin` properties.

Download this sample: `custom-stylesheet.css` and place it at `/var/www/dataverse/branding/custom-stylesheet.css`.

Once you have the location of your custom CSS file, run this curl command to add it to your settings:

```
curl -X PUT -d '/var/www/dataverse/branding/custom-stylesheet.css' http://localhost:8080/api/admin/settings/:StyleCustomizationFile
```

## 4.5.8 Internationalization

Dataverse is being translated into multiple languages by the Dataverse community! Please see below for how to help with this effort!

### Adding Multiple Languages to the Dropdown in the Header

The presence of the `:Languages` database setting adds a dropdown in the header for multiple languages. For example to add English and French to the dropdown:

```
curl http://localhost:8080/api/admin/settings/:Languages -X PUT -d '[{"locale":"en","title":"English"}, {"locale":"fr","title":"Français"}]'
```

### Configuring the “lang” Directory

Translations for Dataverse are stored in “properties” files in a directory on disk (e.g. `/home/glassfish/langBundles`) that you specify with the `dataverse.lang.directory` `dataverse.lang.directory` JVM option, like this:

```
./asadmin create-jvm-options '-Ddataverse.lang.directory=/home/glassfish/langBundles'
```

Go ahead and create the directory you specified.

```
mkdir /home/glassfish/langBundles
```

### Creating a languages.zip File

Dataverse provides an API endpoint for adding languages using a zip file.

First, clone the “dataverse-language-packs” git repo.

```
git clone https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs.git
```

Take a look at <https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs/branches> to see if the version of Dataverse you’re running has translations.

Change to the directory for the git repo you just cloned.

```
cd dataverse-language-packs
```

Switch (`git checkout`) to the branch based on Dataverse version you are running. The branch “dataverse-v4.13” is used in the example below.

```
export BRANCH_NAME=dataverse-v4.13
```

```
git checkout $BRANCH_NAME
```

Create a “languages” directory in “tmp”.

```
mkdir /tmp/languages
```

Copy the properties files into the “languages” directory

```
cp -R en_US/*.properties /tmp/languages
```

```
cp -R fr_CA/*.properties /tmp/languages
```

Create the zip file

```
cd /tmp/languages
```

```
zip languages.zip *.properties
```

### Load the languages.zip file into Dataverse

Now that you have a “languages.zip” file, you can load it into Dataverse with the command below.

```
curl http://localhost:8080/api/admin/datasetfield/loadpropertyfiles -X POST  
--upload-file /tmp/languages/languages.zip -H "Content-Type: application/zip"
```

Click on the languages using the drop down in the header to try them out.

### How to Help Translate Dataverse Into Your Language

Please join the [dataverse-internationalization-wg](https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs) mailing list and contribute to <https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs> to help translate Dataverse into various languages!

Some external tools are also ready to be translated, especially if they are using the `{localeCode}` reserved word in their tool manifest. For details, see the *Building External Tools* section of the API Guide.

## 4.5.9 Web Analytics Code

Your analytics code can be added to your Dataverse installation in a similar fashion to how you brand it, by adding a custom HTML file containing the analytics code snippet and adding the file location to your settings.

Popular analytics providers Google Analytics (<https://www.google.com/analytics/>) and Matomo (formerly “Piwik”; <https://matomo.org/>) have been set up with Dataverse. Use the documentation they provide to add the analytics code to your custom HTML file. This allows for more control of your analytics, making it easier to customize what you prefer to track.

Create your own `analytics-code.html` file using the analytics code snippet provided by Google or Matomo and place it somewhere on the server, outside the application deployment directory; for example: `/var/www/dataverse/branding/analytics-code.html`. Here is an *example* of what your HTML file will look like:

```

<!-- Global Site Tag (gtag.js) - Google Analytics -->
<script async="async" src="https://www.googletagmanager.com/gtag/js?id=YOUR-ACCOUNT-
→CODE"></script>
<script>
  //
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'YOUR-ACCOUNT-CODE');
  //]]&gt;
&lt;/script&gt;
</pre>
</div>
<div data-bbox="111 280 889 326" data-label="Text">
<p><b>IMPORTANT:</b> Note the “async” attribute in the first script line above. In the documentation provided by Google, its value is left blank (as in <code>&lt;script async src="..."&gt;</code>). It must be set as in the example above (<code>&lt;script async="async" src="..."&gt;</code>), otherwise it may cause problems with some browsers.</p>
</div>
<div data-bbox="111 332 889 364" data-label="Text">
<p>Once you have created the analytics file, run this curl command to add it to your settings (using the same file location as in the example above):</p>
</div>
<div data-bbox="111 371 822 401" data-label="Text">
<pre>curl -X PUT -d '/var/www/dataverse/branding/analytics-code.html' http://
localhost:8080/api/admin/settings/:WebAnalyticsCode</pre>
</div>
<div data-bbox="111 425 294 441" data-label="Section-Header">
<h2>Tracking Button Clicks</h2>
</div>
<div data-bbox="111 457 889 517" data-label="Text">
<p>The basic analytics configuration above tracks page navigation. However, it does not capture potentially interesting events, such as those from users clicking buttons on pages, that do not result in a new page opening. In Dataverse, these events include file downloads, requesting access to restricted data, exporting metadata, social media sharing, requesting citation text, launching external tools or WorldMap, contacting authors, and launching computations.</p>
</div>
<div data-bbox="111 524 889 570" data-label="Text">
<p>Both Google and Matomo provide the optional capability to track such events and Dataverse has added CSS style classes (<code>btn-compute</code>, <code>btn-contact</code>, <code>btn-download</code>, <code>btn-explore</code>, <code>btn-export</code>, <code>btn-preview</code>, <code>btn-request</code>, <code>btn-share</code>, and <code>downloadCitation</code>) to its HTML to facilitate it.</p>
</div>
<div data-bbox="111 577 889 637" data-label="Text">
<p>For Google Analytics, the example script at <code>analytics-code.html</code> will track both page hits and events within Dataverse. You would use this file in the same way as the shorter example above, putting it somewhere outside your deployment directory, replacing <code>YOUR ACCOUNT CODE</code> with your actual code and setting <code>:WebAnalyticsCode</code> to reference it.</p>
</div>
<div data-bbox="111 644 889 676" data-label="Text">
<p>Once this script is running, you can look in the Google Analytics console (Realtime/Events or Behavior/Events) and view events by type and/or the Dataset or File the event involves.</p>
</div>
<div data-bbox="111 699 300 718" data-label="Section-Header">
<h3>4.5.10 BagIt Export</h3>
</div>
<div data-bbox="111 733 889 764" data-label="Text">
<p>Dataverse may be configured to submit a copy of published Datasets, packaged as <a href="#">Research Data Alliance conformant</a> zipped <a href="#">BagIt</a> bags to <a href="#">Chronopolis</a> via <a href="#">DuraCloud</a> or alternately to any folder on the local filesystem.</p>
</div>
<div data-bbox="111 770 889 830" data-label="Text">
<p>Dataverse offers an internal archive workflow which may be configured as a PostPublication workflow via an admin API call to manually submit previously published Datasets and prior versions to a configured archive such as Chronopolis. The workflow creates a <a href="#">JSON-LD</a> serialized <a href="#">OAI-ORE</a> map file, which is also available as a metadata export format in the Dataverse web interface.</p>
</div>
<div data-bbox="111 838 889 869" data-label="Text">
<p>At present, the <code>DPNSubmitToArchiveCommand</code> and <code>LocalSubmitToArchiveCommand</code> are the only implementations extending the <code>AbstractSubmitToArchiveCommand</code> and using the configurable mechanisms discussed below.</p>
</div>
<div data-bbox="111 930 264 948" data-label="Page-Footer">4.5. Configuration</div>
<div data-bbox="849 930 889 947" data-label="Page-Footer">227</div>
```

## Duracloud Configuration

Also note that while the current Chronopolis implementation generates the bag and submits it to the archive's DuraCloud interface, the step to make a 'snapshot' of the space containing the Bag (and verify it's successful submission) are actions a curator must take in the DuraCloud interface.

The minimal configuration to support an archiver integration involves adding a minimum of two Dataverse Keys and any required Glassfish jvm options. The example instructions here are specific to the DuraCloud Archiver:

`:ArchiverClassName` - the fully qualified class to be used for archiving. For example:

```
curl http://localhost:8080/api/admin/settings/:ArchiverClassName -X PUT -d
"edu.harvard.iq.dataverse.engine.command.impl.DuraCloudSubmitToArchiveCommand"
```

`:ArchiverSettings` - the archiver class can access required settings including existing Dataverse settings and dynamically defined ones specific to the class. This setting is a comma-separated list of those settings. For example:

```
curl http://localhost:8080/api/admin/settings/:ArchiverSettings -X PUT -d
":DuraCloudHost, :DuraCloudPort, :DuraCloudContext"
```

The DPN archiver defines three custom settings, one of which is required (the others have defaults):

`:DuraCloudHost` - the URL for your organization's Duracloud site. For example:

```
curl http://localhost:8080/api/admin/settings/:DuraCloudHost -X PUT -d "qdr.
duracloud.org"
```

`:DuraCloudPort` and `:DuraCloudContext` are also defined if you are not using the defaults ("443" and "duracloud" respectively). (Note: these settings are only in effect if they are listed in the `:ArchiverSettings`. Otherwise, they will not be passed to the DuraCloud Archiver class.)

Archivers may require glassfish settings as well. For the Chronopolis archiver, the username and password associated with your organization's Chronopolis/DuraCloud account should be configured in Glassfish:

```
./asadmin create-jvm-options '-Dduracloud.username=YOUR_USERNAME_HERE'
./asadmin create-jvm-options '-Dduracloud.password=YOUR_PASSWORD_HERE'
```

## Local Path Configuration

`ArchiverClassName` - the fully qualified class to be used for archiving. For example:

```
curl -X PUT -d "edu.harvard.iq.dataverse.engine.command.impl.
LocalSubmitToArchiveCommand" http://localhost:8080/api/admin/settings/
:ArchiverClassName
```

`:BagItLocalPath` - the path to where you want to store BagIt. For example:

```
curl -X PUT -d /home/path/to/storage http://localhost:8080/api/admin/settings/
:BagItLocalPath
```

`:ArchiverSettings` - the archiver class can access required settings including existing Dataverse settings and dynamically defined ones specific to the class. This setting is a comma-separated list of those settings. For example:

```
curl http://localhost:8080/api/admin/settings/:ArchiverSettings -X PUT -d
":BagItLocalPath"
```

`:BagItLocalPath` is the file path that you've set in `:ArchiverSettings`.



## API Call

Once this configuration is complete, you, as a user with the *PublishDataset* permission, should be able to use the API call to manually submit a *DatasetVersion* for processing:

```
curl -H "X-Dataverse-key: <key>" http://localhost:8080/api/admin/submitDataVersionToArchive/{id}/{version}
```

where:

{id} is the *DatasetId* (or *:persistentId* with the *?persistentId=<DOI>* parameter), and

{version} is the friendly version number, e.g. "1.2".

The *submitDataVersionToArchive* API (and the workflow discussed below) attempt to archive the dataset version via an archive specific method. For Chronopolis, a DuraCloud space named for the dataset (it's DOI with ':' and '.' replaced with '-') is created and two files are uploaded to it: a version-specific *datacite.xml* metadata file and a *BagIt* bag containing the data and an OAI-ORE map file. (The *datacite.xml* file, stored outside the Bag as well as inside is intended to aid in discovery while the ORE map file is 'complete', containing all user-entered metadata and is intended as an archival record.)

In the Chronopolis case, since the transfer from the DuraCloud front-end to archival storage in Chronopolis can take significant time, it is currently up to the admin/curator to submit a 'snap-shot' of the space within DuraCloud and to monitor its successful transfer. Once transfer is complete the space should be deleted, at which point the Dataverse API call can be used to submit a Bag for other versions of the same Dataset. (The space is reused, so that archival copies of different Dataset versions correspond to different snapshots of the same DuraCloud space.)

## PostPublication Workflow

To automate the submission of archival copies to an archive as part of publication, one can setup a Dataverse Workflow using the "archiver" workflow step - see the *Workflows* guide. . The archiver step uses the configuration information discussed above including the *:ArchiverClassName* setting. The workflow step definition should include the set of properties defined in *:ArchiverSettings* in the workflow definition.

To active this workflow, one must first install a workflow using the archiver step. A simple workflow that invokes the archiver step configured to submit to DuraCloud as its only action is included in *dataverse* at */scripts/api/data/workflows/internal-archiver-workflow.json*.

Using the Workflow Native API (see the *Native API* guide) this workflow can be installed using:

```
curl -X POST -H 'Content-type: application/json' --upload-file internal-archiver-workflow.json http://localhost:8080/api/admin/workflows
```

The workflow id returned in this call (or available by doing a GET of */api/admin/workflows* ) can then be submitted as the default PostPublication workflow:

```
curl -X PUT -d {id} http://localhost:8080/api/admin/workflows/default/PostPublishDataset
```

Once these steps are taken, new publication requests will automatically trigger submission of an archival copy to the specified archiver, Chronopolis' DuraCloud component in this example. For Chronopolis, as when using the API, it is currently the admin's responsibility to snap-shot the DuraCloud space and monitor the result. Failure of the workflow, (e.g. if DuraCloud is unavailable, the configuration is wrong, or the space for this dataset already exists due to a prior publication action or use of the API), will create a failure message but will not affect publication itself.

### 4.5.11 Going Live: Launching Your Production Deployment

This guide has attempted to take you from kicking the tires on Dataverse to finalizing your installation before letting real users in. In theory, all this work could be done on a single server but better would be to have separate staging and production environments so that you can deploy upgrades to staging before deploying to production. This “Going Live” section is about launching your **production** environment.

Before going live with your installation of Dataverse, you must take the steps above under “Securing Your Installation” and you should at least review the various configuration options listed below. An attempt has been made to put the more commonly-configured options earlier in the list.

Out of the box, Dataverse attempts to block search engines from crawling your installation of Dataverse so that test datasets do not appear in search results until you’re ready.

#### Letting Search Engines Crawl Your Installation

##### Ensure robots.txt Is Not Blocking Search Engines

For a public production Dataverse installation, it is probably desired that search agents be able to index published pages (AKA - pages that are visible to an unauthenticated user). Polite crawlers usually respect the [Robots Exclusion Standard](#); we have provided an example of a production robots.txt [here](#)).

We **strongly recommend** using the crawler rules in the sample robots.txt linked above. Note that they make the dataverse and dataset pages accessible to the search engine bots; but discourage them from actually crawling the site, by following any search links - facets and such - on the dataverse pages. Such crawling is very inefficient in terms of system resources, and often results in confusing search results for the end users of the search engines (for example, when partial search results are indexed as individual pages).

The recommended solution instead is to directly point the bots to the dataset and dataverse pages that need to be indexed, by advertising them via an explicit sitemap (please see the next section for details on how to generate the sitemap).

You can of course modify your own robots.txt to suit your specific needs as necessary. If you don’t want your datasets to be indexed at all, you can tell the bots to stay away from your site completely. But, as noted above, keep in mind that only the good, “polite” bots honor these rules! You are not really blocking anyone from accessing your site by adding a “Disallow” rule in robots.txt - it is a suggestion only. A rogue bot can and will violate it. If you are having trouble with the site being overloaded with what looks like heavy automated crawling, you may have to resort to blocking this traffic by other means - for example, via rewrite rules in Apache, or even by a Firewall.

(See the sample robots.txt file linked above for some comments on how to set up different “Allow” and “Disallow” rules for different crawler bots)

You have a couple of options for putting an updated robots.txt file into production. If you are fronting Glassfish with Apache as recommended above, you can place robots.txt in the root of the directory specified in your `VirtualHost` and to your Apache config a `ProxyPassMatch` line like the one below to prevent Glassfish from serving the version of robots.txt that is embedded in the Dataverse war file:

```
# don't let Glassfish serve its version of robots.txt
ProxyPassMatch ^/robots.txt$ !
```

For more of an explanation of `ProxyPassMatch` see the [Shibboleth](#) section.

If you are not fronting Glassfish with Apache you’ll need to prevent Glassfish from serving the robots.txt file embedded in the war file by overwriting robots.txt after the war file has been deployed. The downside of this technique is that you will have to remember to overwrite robots.txt in the “exploded” war file each time you deploy the war file, which probably means each time you upgrade to a new version of Dataverse. Furthermore, since the version of Dataverse is always incrementing and the version can be part of the file path, you will need to be

conscious of where on disk you need to replace the file. For example, for Dataverse 4.6.1 the path to robots.txt may be `/usr/local/glassfish4/glassfish/domains/domain1/applications/dataverse-4.6.1/robots.txt` with the version number 4.6.1 as part of the path.

## Creating a Sitemap and Submitting it to Search Engines

Search engines have an easier time indexing content when you provide them a sitemap. The Dataverse sitemap includes URLs to all published dataverses and all published datasets that are not harvested or deaccessioned.

Create or update your sitemap by adding the following curl command to cron to run nightly or as you see fit:

```
curl -X POST http://localhost:8080/api/admin/sitemap
```

This will create or update a file in the following location unless you have customized your installation directory for Glassfish:

```
/usr/local/glassfish4/glassfish/domains/domain1/docroot/sitemap/sitemap.xml
```

On an installation of Dataverse with many datasets, the creation or updating of the sitemap can take a while. You can check Glassfish's server.log file for "BEGIN updateSiteMap" and "END updateSiteMap" lines to know when the process started and stopped and any errors in between.

<https://demo.dataverse.org/sitemap.xml> is the sitemap URL for the Dataverse Demo site and yours should be similar.

Once the sitemap has been generated and placed in the domain docroot directory, it will become available to the outside callers at `<YOUR_SITE_URL>/sitemap/sitemap.xml`; it will also be accessible at `<YOUR_SITE_URL>/sitemap.xml` (via a *pretty-faces* rewrite rule). Some search engines will be able to find it at this default location. Some, **including Google**, need to be **specifically instructed** to retrieve it.

One way to submit your sitemap URL to Google is by using their "Search Console" (<https://search.google.com/search-console>). In order to use the console, you will need to authenticate yourself as the owner of your Dataverse site. Various authentication methods are provided; but if you are already using Google Analytics, the easiest way is to use that account. Make sure you are logged in on Google with the account that has the edit permission on your Google Analytics property; go to the search console and enter the root URL of your Dataverse server, then choose Google Analytics as the authentication method. Once logged in, click on "Sitemaps" in the menu on the left. (todo: add a screenshot?) Consult Google's "submit a sitemap" [instructions](#) for more information; and/or similar instructions for other search engines.

## Putting Your Dataverse Installation on the Map at dataverse.org

Congratulations! You've gone live! It's time to announce your new data repository to the world! You are also welcome to contact [support@dataverse.org](mailto:support@dataverse.org) to have the Dataverse team add your installation to the map at <http://dataverse.org>. Thank you for installing Dataverse!

## Administration of Your Dataverse Installation

Now that you're live you'll want to review the *Admin Guide* for more information about the ongoing administration of a Dataverse installation.

## Setting Up Integrations

Before going live, you might want to consider setting up integrations to make it easier for your users to deposit or explore data. See the *Integrations* section of the Admin Guide for details.

## 4.5.12 JVM Options

JVM stands Java Virtual Machine and as a Java application, Glassfish can read JVM options when it is started. A number of JVM options are configured by the installer below is a complete list of the Dataverse-specific JVM options. You can inspect the configured options by running:

```
./asadmin list-jvm-options | egrep 'dataverse|doi'
```

When changing values these values with `asadmin`, you'll need to delete the old value before adding a new one, like this:

```
./asadmin delete-jvm-options "-Ddataverse.fqdn=old.example.com"
```

```
./asadmin create-jvm-options "-Ddataverse.fqdn=dataverse.example.com"
```

It's also possible to change these values by stopping Glassfish, editing `glassfish4/glassfish/domains/domain1/config/domain.xml`, and restarting Glassfish.

### **dataverse.fqdn**

If the Dataverse server has multiple DNS names, this option specifies the one to be used as the “official” host name. For example, you may want to have `dataverse.example.edu`, and not the less appealing `server-123.socsci.example.edu` to appear exclusively in all the registered global identifiers, Data Deposit API records, etc.

The password reset feature requires `dataverse.fqdn` to be configured.

Do note that whenever the system needs to form a service URL, by default, it will be formed with `https://` and port 443. I.e.,

```
https://{dataverse.fqdn}/
```

If that does not suit your setup, you can define an additional option, `dataverse.siteUrl`, explained below.

### **dataverse.siteUrl**

and specify the protocol and port number you would prefer to be used to advertise the URL for your Dataverse.

For example, configured in `domain.xml`:

```
<jvm-options>-Ddataverse.fqdn=dataverse.example.edu</jvm-options>
```

```
<jvm-options>-Ddataverse.siteUrl=http://{dataverse.fqdn}:8080</jvm-options>
```

### **dataverse.files.directory**

This is how you configure the path to which files uploaded by users are stored.

### **dataverse.auth.password-reset-timeout-in-minutes**

Users have 60 minutes to change their passwords by default. You can adjust this value here.

### **dataverse.rserve.host**

Configuration for *TwoRavens*.

**dataverse.rserve.port**

Configuration for *TwoRavens*.

**dataverse.rserve.user**

Configuration for *TwoRavens*.

**dataverse.rserve.tempdir**

Configuration for *TwoRavens*.

**dataverse.rserve.password**

Configuration for *TwoRavens*.

**dataverse.dropbox.key**

Dropbox provides a Chooser app, which is a Javascript component that allows you to upload files to Dataverse from Dropbox. It is an optional configuration setting, which requires you to pass it an app key and configure the `:UploadMethods` database setting. For more information on setting up your Chooser app, visit <https://www.dropbox.com/developers/chooser>.

```
./asadmin create-jvm-options "-Ddataverse.dropbox.key={{YOUR_APP_KEY}}"
```

**dataverse.path.imagemagick.convert**

For overriding the default path to the `convert` binary from ImageMagick (`/usr/bin/convert`).

**dataverse.dataAccess.thumbnail.image.limit**

For limiting the size (in bytes) of thumbnail images generated from files.

**dataverse.dataAccess.thumbnail.pdf.limit**

For limiting the size (in bytes) of thumbnail images generated from files.

**doi.baseurlstring**

As of this writing, “<https://mds.datacite.org>” (DataCite) and “<https://ezid.cdlib.org>” (EZID) are the main valid values.

While the above two options are recommended because they have been tested by the Dataverse team, it is also possible to use a DataCite Client API as a proxy to DataCite. In this case, requests made to the Client API are captured and passed on to DataCite for processing. The application will interact with the DataCite Client API exactly as if it were interacting directly with the DataCite API, with the only difference being the change to the base endpoint URL.

For example, the Australian Data Archive (ADA) successfully uses the Australian National Data Service (ANDS) API (a proxy for DataCite) to mint their DOIs through Dataverse using a `doi.baseurlstring` value of “<https://researchdata.andes.org.au/api/doi/datacite>” as documented at <https://documentation.andes.org.au/display/DOC/ANDS+DataCite+Client+API>. As ADA did for ANDS DOI minting, any DOI provider (and their corresponding DOI

configuration parameters) other than DataCite must be tested with Dataverse to establish whether or not it will function properly.

Out of the box, Dataverse is configured to use a test MDS DataCite base URL string. You can delete it like this:

```
./asadmin delete-jvm-options '-Ddoi.baseurlstring=https\://mds.test.datacite.org'
```

Then, to switch to production DataCite, you can issue the following command:

```
./asadmin create-jvm-options '-Ddoi.baseurlstring=https\://mds.datacite.org'
```

See also these related database settings below:

- *:DoiProvider*
- *:Protocol*
- *:Authority*
- *:Shoulder*

### **doi.username**

Used in conjunction with `doi.baseurlstring`.

Once you have a username from your provider, you can enter it like this:

```
./asadmin create-jvm-options '-Ddoi.username=YOUR_USERNAME_HERE'
```

### **doi.password**

Used in conjunction with `doi.baseurlstring`.

Once you have a password from your provider, you can enter it like this:

```
./asadmin create-jvm-options '-Ddoi.password=YOUR_PASSWORD_HERE'
```

### **dataverse.handlenet.admcredfile**

If you're using **handles**, this JVM setting configures access credentials so your dataverse can talk to your Handle.Net server. This is the private key generated during Handle.Net server installation. Typically the full path is set to `handle/svr_1/admpriv.bin`. Please refer to [Handle.Net's documentation](#) for more info.

### **dataverse.handlenet.admprivphrase**

This JVM setting is also part of **handles** configuration. The Handle.Net installer lets you choose whether to encrypt the `admcredfile` private key or not. If you do encrypt it, this is the pass phrase that it's encrypted with.

### **dataverse.handlenet.index**

If you want to use different index than the default 300

### **dataverse.timerServer**

This JVM option is only relevant if you plan to run multiple Glassfish servers for redundancy. Only one Glassfish server can act as the dedicated timer server and for details on promoting or demoting a Glassfish server to handle this responsibility, see *Dataverse Application Timers*.

### **dataverse.lang.directory**

This JVM option is used to configure the path where all the language specific property files are to be stored. If this option is set then the English property file must be present in the path along with any other language property file. You can download language property files from <https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs>

```
./asadmin create-jvm-options '-Ddataverse.lang.directory=PATH_LOCATION_HERE'
```

If this value is not set, by default, a Dataverse installation will read the English language property files from the Java Application.

See also *Internationalization*.

### **dataverse.files.hide-schema-dot-org-download-urls**

Please note that this setting is experimental.

By default, download URLs to files will be included in Schema.org JSON-LD output. To prevent these URLs from being included in the output, set `dataverse.files.hide-schema-dot-org-download-urls` to true as in the example below.

```
./asadmin create-jvm-options '-Ddataverse.files.hide-schema-dot-org-download-urls=true'
```

Please note that there are other reasons why download URLs may not be included for certain files such as if a guestbook entry is required or if the file is restricted.

For more on Schema.org JSON-LD, see the *Metadata Export* section of the Admin Guide.

## **4.5.13 Database Settings**

These settings are stored in the `setting` database table but can be read and modified via the “admin” endpoint of the *Native API* for easy scripting.

The most commonly used configuration options are listed first.

The pattern you will observe in curl examples below is that an HTTP PUT is used to add or modify a setting. If you perform an HTTP GET (the default when using curl), the output will contain the value of the setting, if it has been set. You can also do a GET of all settings with `curl http://localhost:8080/api/admin/settings` which you may want to pretty-print by piping the output through a tool such as `jq` by appending `| jq ..` If you want to remove a setting, use an HTTP DELETE such as `curl -X DELETE http://localhost:8080/api/admin/settings/:GuidesBaseUrl`.

### **:BlockedApiPolicy**

Out of the box, all API endpoints are completely open, as mentioned in the section on security above. It is highly recommended that you choose one of the policies below and also configure `:BlockedApiEndpoints`.

- `localhost-only`: Allow from localhost.

- `unlock-key`: Require a key defined in `:BlockedApiKey`.
- `drop`: Disallow the blocked endpoints completely.

```
curl -X PUT -d localhost-only http://localhost:8080/api/admin/settings/  
:BlockedApiPolicy
```

### **:BlockedApiEndpoints**

A comma separated list of API endpoints to be blocked. For a production installation, “admin” should be blocked (and perhaps “builtin-users” as well), as mentioned in the section on security above:

```
curl -X PUT -d "admin,builtin-users" http://localhost:8080/api/admin/settings/  
:BlockedApiEndpoints
```

See the *API Guide* for a list of API endpoints.

### **:BlockedApiKey**

Used in conjunction with the `:BlockedApiPolicy` being set to `unlock-key`. When calling blocked APIs, add a query parameter of `unlock-key=theKeyYouChose` to use the key.

```
curl -X PUT -d s3kretKey http://localhost:8080/api/admin/settings/  
:BlockedApiKey
```

### **BuiltinUsers.KEY**

The key required to create users via API as documented at *Native API*. Unlike other database settings, this one doesn’t start with a colon.

```
curl -X PUT -d builtinS3kretKey http://localhost:8080/api/admin/settings/  
BuiltinUsers.KEY
```

### **:SearchApiRequiresToken**

In Dataverse 4.7 and lower, the *Search API* required an API token, but as of Dataverse 4.7.1 this is no longer the case. If you prefer the old behavior of requiring API tokens to use the Search API, set `:SearchApiRequiresToken` to `true`.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/  
:SearchApiRequiresToken
```

### **:SystemEmail**

This is the email address that “system” emails are sent from such as password reset links. Your Dataverse installation will not send mail without this setting in place.

```
curl -X PUT -d 'LibraScholar SWAT Team <support@librascholar.edu>' http://  
localhost:8080/api/admin/settings/:SystemEmail
```

Note that only the email address is required, which you can supply without the `<` and `>` signs, but if you include the text, it’s the way to customize the name of your support team, which appears in the “from” address in emails as well as in help text in the UI.

Please note that if you’re having any trouble sending email, you can refer to “Troubleshooting” under *Installation*.



**:HomePageCustomizationFile**

See *Branding Your Installation* above.

**:LogoCustomizationFile**

See *Branding Your Installation* above.

**:HeaderCustomizationFile**

See *Branding Your Installation* above.

**:DisableRootDataverseTheme**

See *Branding Your Installation* above.

**:FooterCustomizationFile**

See *Branding Your Installation* above.

**:StyleCustomizationFile**

See *Branding Your Installation* above.

**:WebAnalyticsCode**

See *Web Analytics Code* above.

**:FooterCopyright**

By default the footer says “Copyright © [YYYY]” but you can add text after the year, as in the example below.

```
curl -X PUT -d ", Your Institution" http://localhost:8080/api/admin/settings/
:FooterCopyright
```

**:DoiProvider**

As of this writing “DataCite” and “EZID” are the only valid options for production installations. Developers using version 4.10 and above are welcome to use the keyword “FAKE” to configure a non-production installation with a non-resolving, in-code provider, which will basically short-circuit the DOI publishing process. `:DoiProvider` is only needed if you are using DOI.

```
curl -X PUT -d DataCite http://localhost:8080/api/admin/settings/:DoiProvider
```

This setting relates to the `:Protocol`, `:Authority`, `:Shoulder`, and `:IdentifierGenerationStyle` database settings below as well as the following JVM options:

- `doi.baseurlstring`
- `doi.username`
- `doi.password`

### :Protocol

As of this writing “doi” and “hdl” are the only valid option for the protocol for a persistent ID.

```
curl -X PUT -d doi http://localhost:8080/api/admin/settings/:Protocol
```

### :Authority

Use the authority assigned to you by your DoiProvider or HandleProvider.

Please note that the authority cannot have a slash (“/”) in it.

```
curl -X PUT -d 10.xxxx http://localhost:8080/api/admin/settings/:Authority
```

### :Shoulder

Out of the box, the DOI shoulder is set to “FK2/” but this is for testing only! When you apply for your DOI namespace, you may have requested a shoulder. The following is only an example and a trailing slash is optional.

```
curl -X PUT -d "MyShoulder/" http://localhost:8080/api/admin/settings/
:Shoulder
```

### :IdentifierGenerationStyle

By default, Dataverse generates a random 6 character string, pre-pended by the Shoulder if set, to use as the identifier for a Dataset. Set this to `sequentialNumber` to use sequential numeric values instead (again pre-pended by the Shoulder if set). (the assumed default setting is `randomString`). In addition to this setting, a database sequence must be created in the database. We provide the script below (downloadable [here](#)). You may need to make some changes to suit your system setup, see the comments for more information:

```
-- A script for creating a numeric identifier sequence, and an external
-- stored procedure, for accessing the sequence from inside the application,
-- in a non-hacky, JPA way.

-- NOTE:

-- 1. The database user name "dvnapp" is hard-coded here - it may
-- need to be changed to match your database user name;

-- 2. In the code below, the sequence starts with 1, but it can be adjusted by
-- changing the MINVALUE as needed.

CREATE SEQUENCE datasetidentifier_seq
  INCREMENT 1
  MINVALUE 1
  MAXVALUE 9223372036854775807
  START 1
  CACHE 1;

ALTER TABLE datasetidentifier_seq OWNER TO "dvnapp";

-- And now create a PostgreSQL FUNCTION, for JPA to
-- access as a NamedStoredProcedure:

CREATE OR REPLACE FUNCTION generateIdentifierAsSequentialNumber(
  OUT identifier int)
```

```

    RETURNS int AS
$BODY$
BEGIN
    select nextval('datasetidentifier_seq') into identifier;
END;
$BODY$
LANGUAGE plpgsql;

```

Note that the SQL above is Postgres-specific. If necessary, it can be reimplemented in any other SQL flavor - the standard JPA code in the application simply expects the database to have a saved function (“stored procedure”) named `generateIdentifierAsSequentialNumber` with the single return argument `identifier`.

Please note that `:IdentifierGenerationStyle` also plays a role for the “identifier” for files. See the section on `:DataFilePIDFormat` below for more details.

### **:DataFilePIDFormat**

This setting controls the way that the “identifier” component of a file’s persistent identifier (PID) relates to the PID of its “parent” dataset.

By default the identifier for a file is dependent on its parent dataset. For example, if the identifier of a dataset is “TJCLKP”, the identifier for a file within that dataset will consist of the parent dataset’s identifier followed by a slash (“/”), followed by a random 6 character string, yielding “TJCLKP/MLGWJO”. Identifiers in this format are what you should expect if you leave `:DataFilePIDFormat` undefined or set it to `DEPENDENT` and have not changed the `:IdentifierGenerationStyle` setting from its default.

Alternatively, the identifier for File PIDs can be configured to be independent of Dataset PIDs using the setting “`INDEPENDENT`”. In this case, file PIDs will not contain the PIDs of their parent datasets, and their PIDs will be generated the exact same way that datasets’ PIDs are, based on the `:IdentifierGenerationStyle` setting described above (random 6 character strings or sequential numbers, pre-pended by any shoulder).

The chart below shows examples from each possible combination of parameters from the two settings. `:IdentifierGenerationStyle` can be either `randomString` (the default) or `sequentialNumber` and `:DataFilePIDFormat` can be either `DEPENDENT` (the default) or `INDEPENDENT`. In the examples below the “identifier” for the dataset is “TJCLKP” for “`randomString`” and “100001” for “`sequentialNumber`”.

	<code>randomString</code>	<code>sequentialNumber</code>
<b>DEPENDENT</b>	TJCLKP/MLGWJO	100001/1
<b>INDEPENDENT</b>	MLGWJO	100002

As seen above, in cases where `:IdentifierGenerationStyle` is set to `sequentialNumber` and `:DataFilePIDFormat` is set to `DEPENDENT`, each file within a dataset will be assigned a number *within* that dataset starting with “1”.

Otherwise, if `:DataFilePIDFormat` is set to `INDEPENDENT`, then each file will be assigned a PID with the next number in the overall sequence, regardless of what dataset it is in. If the file is created after a dataset with the PID 100001, then the file will be assigned the PID 100002. This option is functional, but it is not a recommended use case.

Note that in either case, when using the `sequentialNumber` option, datasets and files share the same database sequence that was created as part of the setup described in `:IdentifierGenerationStyle` above.

### **:FilePIDsEnabled**

Toggles publishing of file-based PIDs for the entire installation. By default this setting is absent and Dataverse assumes it to be true.

If you don’t want to register file-based PIDs for your installation, set:

```
curl -X PUT -d 'false' http://localhost:8080/api/admin/settings/
:FilePIDsEnabled
```

Note: File-level PID registration was added in 4.9 and is required until version 4.9.3.

### **:IndependentHandleService**

Specific for Handle PIDs. Set this setting to true if you want to use a Handle service which is setup to work ‘independently’ (No communication with the Global Handle Registry). By default this setting is absent and Dataverse assumes it to be false.

```
curl -X PUT -d 'true' http://localhost:8080/api/admin/settings/
:IndependentHandleService
```

### **:ApplicationTermsOfUse**

Upload an default language HTML file containing the Terms of Use to be displayed at sign up. Supported HTML tags are listed under the *Dataset + File Management* section of the User Guide.

```
curl -X PUT -d@/tmp/apptou.html http://localhost:8080/api/admin/settings/
:ApplicationTermsOfUse
```

To upload a language specific Terms of Use file,

```
curl -X PUT -d@/tmp/apptou_fr.html http://localhost:8080/api/admin/settings/
:ApplicationTermsOfUse/lang/fr
```

To delete language specific option,

```
curl -X DELETE http://localhost:8080/api/admin/settings/:ApplicationTermsOfUse/
lang/fr
```

Unfortunately, in most cases, the text file will probably be too big to upload (>1024 characters) due to a bug. A workaround has been posted to <https://github.com/IQSS/dataverse/issues/2669>

### **:ApplicationPrivacyPolicyUrl**

Specify a URL where users can read your Privacy Policy, linked from the bottom of the page.

```
curl -X PUT -d https://dataverse.org/best-practices/harvard-dataverse-privacy-policy
http://localhost:8080/api/admin/settings/:ApplicationPrivacyPolicyUrl
```

### **:ApiTermsOfUse**

Specify a URL where users can read your API Terms of Use. API users can retrieve this URL from the SWORD Service Document or the “info” section of our *Native API* documentation.

```
curl -X PUT -d https://dataverse.org/best-practices/harvard-api-tou http://
localhost:8080/api/admin/settings/:ApiTermsOfUse
```

### **:ExcludeEmailFromExport**

See also *Privacy Considerations* above.

Set `:ExcludeEmailFromExport` to prevent email addresses for contacts from being exposed in XML or JSON representations of dataset and dataverse metadata. For a list exported formats such as DDI, see the *Metadata Export* section of the Admin Guide.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/
:ExcludeEmailFromExport
```

### **:NavbarAboutUrl**

Set `NavbarAboutUrl` to a fully-qualified URL which will be used for the “About” link in the navbar.

Note: The “About” link will not appear in the navbar until this option is set.

```
curl -X PUT -d http://dataverse.example.edu http://localhost:8080/api/admin/
settings/:NavbarAboutUrl
```

### **:GuidesBaseUrl**

Set `GuidesBaseUrl` to override the default value “<http://guides.dataverse.org>”. If you are interested in writing your own version of the guides, you may find the *Writing Documentation* section of the Developer Guide helpful.

```
curl -X PUT -d http://dataverse.example.edu http://localhost:8080/api/admin/
settings/:GuidesBaseUrl
```

### **:GuidesVersion**

Set `:GuidesVersion` to override the version number in the URL of guides. For example, rather than <http://guides.dataverse.org/en/4.6/user/account.html> the version is overridden to <http://guides.dataverse.org/en/1234-new-feature/user/account.html> in the example below:

```
curl -X PUT -d 1234-new-feature http://localhost:8080/api/admin/settings/
:GuidesVersion
```

### **:NavbarSupportUrl**

Set `:NavbarSupportUrl` to a fully-qualified URL which will be used for the “Support” link in the navbar.

Note that this will override the default behaviour for the “Support” menu option, which is to display the dataverse ‘feedback’ dialog.

```
curl -X PUT -d http://dataverse.example.edu/supportpage.html http://
localhost:8080/api/admin/settings/:NavbarSupportUrl
```

### **:MetricsUrl**

Make the metrics component on the root dataverse a clickable link to a website where you present metrics on your Dataverse installation, perhaps one of the community-supported tools mentioned in the *Reporting Tools* section of the Admin Guide.

```
curl -X PUT -d http://metrics.dataverse.example.edu http://localhost:8080/api/
admin/settings/:MetricsUrl
```

### **:StatusMessageHeader**

For dynamically adding an informational header to the top of every page. `StatusMessageText` must also be set for a message to show. For example, “For testing only...” at the top of <https://demo.dataverse.org> is set with this:

```
curl -X PUT -d "For testing only..." http://localhost:8080/api/admin/settings/
:StatusMessageHeader
```

You can make the text clickable and include an additional message in a pop up by setting `:StatusMessageText`.

### **:StatusMessageText**

Alongside the `:StatusMessageHeader` you need to add `StatusMessageText` for the message to show.:

```
curl -X PUT -d "This appears in a popup." http://localhost:8080/api/admin/
settings/:StatusMessageText
```

### **:MaxFileUploadSizeInBytes**

Set `MaxFileUploadSizeInBytes` to “2147483648”, for example, to limit the size of files uploaded to 2 GB.

Notes:

- For SWORD, this size is limited by the Java Integer.MAX\_VALUE of 2,147,483,647. (see: <https://github.com/IQSS/dataverse/issues/2169>)
- If the `MaxFileUploadSizeInBytes` is NOT set, uploads, including SWORD may be of unlimited size.
- For larger file upload sizes, you may need to configure your reverse proxy timeout. If using apache2 (httpd) with Shibboleth, add a timeout to the ProxyPass defined in `etc/httpd/conf.d/ssl.conf` (which is described in the *Shibboleth* setup).

```
curl -X PUT -d 2147483648 http://localhost:8080/api/admin/settings/
:MaxFileUploadSizeInBytes
```

### **:ZipDownloadLimit**

For performance reasons, Dataverse will only create zip files on the fly up to 100 MB but the limit can be increased. Here’s an example of raising the limit to 1 GB:

```
curl -X PUT -d 1000000000 http://localhost:8080/api/admin/settings/
:ZipDownloadLimit
```

### **:TabularIngestSizeLimit**

Threshold in bytes for limiting whether or not “ingest” it attempted for tabular files (which can be resource intensive). For example, with the below in place, files greater than 2 GB in size will not go through the ingest process:

```
curl -X PUT -d 2000000000 http://localhost:8080/api/admin/settings/
:TabularIngestSizeLimit
```

(You can set this value to 0 to prevent files from being ingested at all.)

You can override this global setting on a per-format basis for the following formats:

- DTA
- POR

- SAV
- Rdata
- CSV
- XLSX

For example, if you want your installation of Dataverse to not attempt to ingest Rdata files larger than 1 MB, use this setting:

```
curl -X PUT -d 1000000 http://localhost:8080/api/admin/settings/  
:TabularIngestSizeLimit:Rdata
```

### **:ZipUploadFilesLimit**

Limit the number of files in a zip that Dataverse will accept.

### **:SolrHostColonPort**

By default Dataverse will attempt to connect to Solr on port 8983 on localhost. Use this setting to change the hostname or port. You must restart Glassfish after making this change.

```
curl -X PUT -d localhost:8983 http://localhost:8080/api/admin/settings/  
:SolrHostColonPort
```

### **:SolrFullTextIndexing**

Whether or not to index the content of files such as PDFs. The default is false.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/  
:SolrFullTextIndexing
```

### **:SolrMaxFileSizeForFullTextIndexing**

If `:SolrFullTextIndexing` is set to true, the content of files of any size will be indexed. To set a limit in bytes for which files to index in this way:

```
curl -X PUT -d 314572800 http://localhost:8080/api/admin/settings/  
:SolrMaxFileSizeForFullTextIndexing
```

### **:SignUpUrl**

The relative path URL to which users will be sent for signup. The default setting is below.

```
curl -X PUT -d '/dataverseuser.xhtml?editMode=CREATE' http://localhost:8080/  
api/admin/settings/:SignUpUrl
```

### **:LoginSessionTimeout**

Session timeout (in minutes) for logged-in users. The default is 8 hours (480 minutes). For the anonymous user sessions, the timeout is set to the default value, configured in the web.xml file of the Dataverse application.

In the example below we reduce the timeout to 4 hours:

```
curl -X PUT -d 240 http://localhost:8080/api/admin/settings/
:LoginSessionTimeout
```

### **:TwoRavensUrl**

The `:TwoRavensUrl` option is no longer valid. See *TwoRavens* and the *External Tools* section of the Admin Guide.

### **:TwoRavensTabularView**

The `:TwoRavensTabularView` option is no longer valid. See *TwoRavens* and the *External Tools* section of the Admin Guide.

### **:GeoconnectCreateEditMaps**

Set `GeoconnectCreateEditMaps` to true to allow the user to create GeoConnect Maps. This boolean effects whether the user sees the map button on the dataset page and if the ingest will create a shape file.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/
:GeoconnectCreateEditMaps
```

### **:GeoconnectViewMaps**

Set `GeoconnectViewMaps` to true to allow a user to view existing maps. This boolean effects whether a user will see the “Explore” button.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/
:GeoconnectViewMaps
```

### **:DatasetPublishPopupCustomText**

Set custom text a user will view when publishing a dataset. Note that this text is exposed via the “Info” endpoint of the *Native API*.

```
curl -X PUT -d "Deposit License Requirements" http://localhost:8080/api/admin/
settings/:DatasetPublishPopupCustomText
```

If you have a long text string, you can upload it as a file as in the example below.

```
curl -X PUT --upload-file /tmp/long.txt http://localhost:8080/api/admin/
settings/:DatasetPublishPopupCustomText
```

### **:DatasetPublishPopupCustomTextOnAllVersions**

Set whether a user will see the custom text when publishing all versions of a dataset

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/
:DatasetPublishPopupCustomTextOnAllVersions
```



### **:SearchHighlightFragmentSize**

Set `SearchHighlightFragmentSize` to override the default value of 100 from <https://wiki.apache.org/solr/HighlightingParameters#hl.fragsize>. In practice, a value of “320” seemed to fix the issue at <https://github.com/IQSS/dataverse/issues/2191>

```
curl -X PUT -d 320 http://localhost:8080/api/admin/settings/
:SearchHighlightFragmentSize
```

### **:ScrubMigrationData**

Allow for migration of non-conformant data (especially dates) from DVN 3.x to Dataverse 4.

### **:MinutesUntilConfirmEmailTokenExpires**

The duration in minutes before “Confirm Email” URLs expire. The default is 1440 minutes (24 hours). See also the *User Administration* section of our Admin Guide.

### **:DefaultAuthProvider**

If you have enabled Shibboleth and/or one or more OAuth providers, you may wish to make one of these authentication providers the default when users visit the Log In page. If unset, this will default to `builtin` but these valid options (depending if you’ve done the setup described in the *Shibboleth* or *OAuth Login: ORCID, GitHub, Google* sections) are:

- `builtin`
- `shib`
- `orcid`
- `github`
- `google`

Here is an example of setting the default auth provider back to `builtin`:

```
curl -X PUT -d builtin http://localhost:8080/api/admin/settings/
:DefaultAuthProvider
```

### **:AllowSignUp**

Set to false to disallow local accounts to be created. See also the sections on *Shibboleth* and *OAuth Login: ORCID, GitHub, Google*.

### **:FileFixityChecksumAlgorithm**

Dataverse calculates checksums for uploaded files so that users can determine if their file was corrupted via upload or download. This is sometimes called “file fixity”: [https://en.wikipedia.org/wiki/File\\_Fixity](https://en.wikipedia.org/wiki/File_Fixity)

The default checksum algorithm used is MD5 and should be sufficient for establishing file fixity. “SHA-1”, “SHA-256” and “SHA-512” are alternate values for this setting. For example:

```
curl -X PUT -d 'SHA-512' http://localhost:8080/api/admin/settings/
:FileFixityChecksumAlgorithm
```

The fixity algorithm used on existing files can be changed by a superuser using the API. An optional query parameter (num) can be used to limit the number of updates attempted. The API call will only update the algorithm and checksum for a file if the existing checksum can be validated against the file. Statistics concerning the updates are returned in the response to the API call with details in the log.

```
curl http://localhost:8080/api/admin/updateHashValues/{alg}      curl http://localhost:8080/api/admin/updateHashValues/{alg}?num=1
```

### **:PVMinLength**

Password policy setting for builtin user accounts: a password's minimum valid character length. The default is 6.

```
curl -X PUT -d 6 http://localhost:8080/api/admin/settings/:PVMinLength
```

### **:PVMaxLength**

Password policy setting for builtin user accounts: a password's maximum valid character length.

```
curl -X PUT -d 0 http://localhost:8080/api/admin/settings/:PVMaxLength
```

### **:PVNumberOfConsecutiveDigitsAllowed**

By default, passwords can contain an unlimited number of digits in a row. However, if your password policy specifies otherwise (e.g. only four digits in a row are allowed), then you can issue the following curl command to set the number of consecutive digits allowed (this example uses 4):

```
curl -X PUT -d 4 http://localhost:8080/api/admin/settings/:PVNumberOfConsecutiveDigitsAllowed
```

### **:PVCharacterRules**

Password policy setting for builtinuser accounts: dictates which types of characters can be required in a password. This setting goes hand-in-hand with *:PVNumberOfCharacteristics*. The default setting contains two rules:

- one letter
- one digit

The default setting above is equivalent to specifying “Alphabetical:1,Digit:1”.

By specifying “UpperCase:1,LowerCase:1,Digit:1,Special:1”, for example, you can put the following four rules in place instead:

- one uppercase letter
- one lowercase letter
- one digit
- one special character

If you have implemented 4 different character rules in this way, you can also optionally increase *:PVNumberOfCharacteristics* to as high as 4. However, please note that *:PVNumberOfCharacteristics* cannot be set to a number higher than the number of character rules or you will see the error, “Number of characteristics must be <= to the number of rules”.

Also note that the Alphabetical setting should not be used in tandem with the UpperCase or LowerCase settings. The Alphabetical setting encompasses both of those more specific settings, so using it with them will cause your password policy to be unnecessarily confusing, and potentially easier to bypass.

```
curl -X PUT -d 'UpperCase:1,LowerCase:1,Digit:1,Special:1' http://localhost:8080/api/admin/settings/:PVCharacterRules
```

```
curl -X PUT -d 3 http://localhost:8080/api/admin/settings/:PVNumberOfCharacteristics
```

### **:PVNumberOfCharacteristics**

Password policy setting for builtin user accounts: the number indicates how many of the character rules defined by :PVCharacterRules are required as part of a password. The default is 2. :PVNumberOfCharacteristics cannot be set to a number higher than the number of rules or you will see the error, “Number of characteristics must be <= to the number of rules”.

```
curl -X PUT -d 2 http://localhost:8080/api/admin/settings/:PVNumberOfCharacteristics
```

### **:PVDictionaries**

Password policy setting for builtin user accounts: set a comma separated list of dictionaries containing words that cannot be used in a user password. /usr/share/dict/words is suggested and shown modified below to not contain words 3 letters or less. You are free to choose a different dictionary. By default, no dictionary is checked.

```
DIR=THE_PATH_YOU_WANT_YOUR_DICTIONARY_TO_RESIDE sed '/^\.{3}$/d' /usr/share/dict/words > $DIR/pwdictionary curl -X PUT -d "$DIR/pwdictionary" http://localhost:8080/api/admin/settings/:PVDictionaries
```

### **:PVGoodStrength**

Password policy setting for builtin user accounts: passwords of equal or greater character length than the :PVGoodStrength setting are always valid, regardless of other password constraints.

```
curl -X PUT -d 20 http://localhost:8080/api/admin/settings/:PVGoodStrength
```

Recommended setting: 20.

### **:PVCustomPasswordResetAlertMessage**

Changes the default info message displayed when a user is required to change their password on login. The default is:

```
{0} Reset Password{1} - Our password requirements have changed. Please pick a strong password that matches the criteria below.
```

Where the {0} and {1} denote surrounding HTML **bold** tags. It's recommended to put a single space before your custom message for better appearance (as in the default message above). Including the {0} and {1} to bolden part of your message is optional.

Customize the message using the following curl command's syntax:

```
curl -X PUT -d '{0} Action Required:{1} Your current password does not meet all requirements. Please enter a new password meeting the criteria below.' http://localhost:8080/api/admin/settings/:PVCustomPasswordResetAlertMessage
```

### **:ShibPassiveLoginEnabled**

Set `:ShibPassiveLoginEnabled` to `true` to enable passive login for Shibboleth. When this feature is enabled, an additional Javascript file (`isPassive.js`) will be loaded for every page. It will generate a passive login request to your Shibboleth SP when an anonymous user navigates to the site. A cookie named “`_check_is_passive_dv`” will be created to keep track of whether or not a passive login request has already been made for the user.

This implementation follows the example on the Shibboleth wiki documentation page for the `isPassive` feature: <https://wiki.shibboleth.net/confluence/display/SHIB2/isPassive>

It is recommended that you configure additional error handling for your Service Provider if you enable passive login. A good way of doing this is described in the Shibboleth wiki documentation:

- *In your Service Provider 2.x `shibboleth2.xml` file, add `redirectErrors="#THIS PAGE#"` to the `Errors` element.*

You can set the value of “`#THIS PAGE#`” to the URL of your Dataverse homepage, or any other page on your site that is accessible to anonymous users and will have the `isPassive.js` file loaded.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/
:ShibPassiveLoginEnabled
```

### **:ComputeBaseUrl**

Set the base URL for the “Compute” button for a dataset.

```
curl -X PUT -d 'https://giji.massopencloud.org/application/dataverse' http://
localhost:8080/api/admin/settings/:ComputeBaseUrl
```

### **:CloudEnvironmentName**

Set the name of the cloud environment you’ve integrated with your Dataverse installation.

```
curl -X PUT -d 'Massachusetts Open Cloud (MOC)' http://localhost:8080/api/
admin/settings/:CloudEnvironmentName
```

### **:PublicInstall**

Setting an installation to public will remove the ability to restrict data files or datasets. This functionality of Dataverse will be disabled from your installation.

This is useful for specific cases where an installation’s files are stored in public access. Because files stored this way do not obey Dataverse’s file restrictions, users would still be able to access the files even when they’re restricted. In these cases it’s best to use `:PublicInstall` to disable the feature altogether.

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/:PublicInstall
```

### **:DataCaptureModuleUrl**

The URL for your Data Capture Module (DCM) installation. This component is experimental and can be downloaded from <https://github.com/sbgrid/data-capture-module> .

```
curl -X PUT -d 'https://dcm.example.edu' http://localhost:8080/api/admin/
settings/:DataCaptureModuleUrl
```

### **:RepositoryStorageAbstractionLayerUrl**

The URL for your Repository Storage Abstraction Layer (RSAL) installation. This component is experimental and can be downloaded from <https://github.com/sbgrid/rsal>.

```
curl -X PUT -d 'https://rsal.example.edu' http://localhost:8080/api/admin/
settings/:RepositoryStorageAbstractionLayerUrl
```

### **:UploadMethods**

This setting controls which upload methods are available to users of your installation of Dataverse. The following upload methods are available:

- `native/http`: Corresponds to “Upload with HTTP via your browser” and APIs that use HTTP (SWORD and native).
- `dcm/rsync+ssh`: Corresponds to “Upload with rsync+ssh via Data Capture Module (DCM)”. A lot of setup is required, as explained in the *Big Data Support* section of the Dev Guide.

Out of the box only `native/http` is enabled and will work without further configuration. To add multiple upload method, separate them using a comma like this:

```
curl -X PUT -d 'native/http,dcm/rsync+ssh' http://localhost:8080/api/admin/
settings/:UploadMethods
```

You’ll always want at least one upload method, so the easiest way to remove one of them is to simply PUT just the one you want, like this:

```
curl -X PUT -d 'native/http' http://localhost:8080/api/admin/settings/
:UploadMethods
```

### **:DownloadMethods**

This setting is experimental and related to Repository Storage Abstraction Layer (RSAL).

```
curl -X PUT -d 'rsal/rsync' http://localhost:8080/api/admin/settings/
:DownloadMethods
```

### **:GuestbookResponsesPageDisplayLimit**

Limit on how many guestbook entries to display on the `guestbook-responses` page. By default, only the 5000 most recent entries will be shown. Use the standard settings API in order to change the limit. For example, to set it to 10,000, make the following API call:

```
curl -X PUT -d 10000 http://localhost:8080/api/admin/settings/
:GuestbookResponsesPageDisplayLimit
```

### **:CustomDatasetSummaryFields**

You can replace the default dataset metadata fields that are displayed above files table on the dataset page with a custom list separated by commas using the curl command below.

```
curl http://localhost:8080/api/admin/settings/:CustomDatasetSummaryFields -X
PUT -d 'producer,subtitle,alternativeTitle'
```

You have to put the `datasetFieldType` name attribute in the `:CustomDatasetSummaryFields` setting for this to work.

### **:AllowApiTokenLookupViaApi**

Dataverse 4.8.1 and below allowed API Token lookup via API but for better security this has been disabled by default. Set this to true if you really want the old behavior.

```
curl -X PUT -d 'true' http://localhost:8080/api/admin/settings/
:AllowApiTokenLookupViaApi
```

### **:ProvCollectionEnabled**

Enable the collection of provenance metadata on Dataverse via the provenance popup.

```
curl -X PUT -d 'true' http://localhost:8080/api/admin/settings/
:ProvCollectionEnabled
```

### **:MetricsCacheTimeoutMinutes**

Sets how long a cached metrics result is used before re-running the query for a request. This timeout is only applied to some of the metrics that query the current state of the system, previous months queries are cached indefinitely. See [Metrics API](#) for more info. The default timeout value is 7 days (10080 minutes).

```
curl -X PUT -d 10080 http://localhost:8080/api/admin/settings/
:MetricsCacheTimeoutMinutes
```

### **:MDCLogPath**

Sets the path where the raw Make Data Count logs are stored before being processed. If not set, no logs will be created for Make Data Count. See also the [Make Data Count](#) section of the Admin Guide.

```
curl -X PUT -d '/usr/local/glassfish4/glassfish/domains/domain1/logs' http://
localhost:8080/api/admin/settings/:MDCLogPath
```

### **:Languages**

Sets which languages should be available. If there is more than one, a dropdown is displayed in the header.

See [Internationalization](#) for a curl example and related settings.

### **:InheritParentRoleAssignments**

`:InheritParentRoleAssignments` can be set to a comma-separated list of role aliases or `*` (all) to cause newly created Dataverses to inherit the set of users and/or internal groups who have assignments for those role(s) on the parent Dataverse, i.e. those users/groups will be assigned the same role(s) on the new Dataverse (in addition to the creator of the new Dataverse having an admin role). This can be helpful in situations where multiple organizations are sharing one Dataverse instance. The default, if `::InheritParentRoleAssignments` is not set is for the creator of the new Dataverse to be the only one assigned a role.

```
curl -X PUT -d 'admin, curator' http://localhost:8080/api/admin/settings/
:InheritParentRoleAssignments or curl -X PUT -d '*' http://localhost:8080/api/
admin/settings/:InheritParentRoleAssignments
```

### :AllowCors

Allows Cross-Origin Resource sharing(CORS). By default this setting is absent and Dataverse assumes it to be true.

If you don't want to allow CORS for your installation, set:

```
curl -X PUT -d 'false' http://localhost:8080/api/admin/settings/:AllowCors
```

## 4.6 Upgrading

When upgrading within Dataverse 4.x, you will need to follow the upgrade instructions for each intermediate version.

Upgrades always involve deploying the latest war file but may also include updating the schema used by Solr or other manual steps. Running database migration scripts was once required but this has been automated (see the `flyway_schema_history` database table to see migrations that have been run).

Please consult the release notes associated with each release at <https://github.com/IQSS/dataverse/releases> for more information.

Upgrading from DVN 3.x is actually a migration due to the many changes. Migration scripts have been checked into the source tree but as of this writing it is expected that people will require assistance running them. Please reach out per the *Introduction* section.

## 4.7 TwoRavens

TwoRavens is a web application for tabular data exploration and statistical analysis. It can be integrated with Dataverse, as an **optional** component. While TwoRavens was originally created at IQSS, its developers have since left the organization. Plans for the future of the Dataverse/TwoRavens collaboration are still being worked out. As such, **support for TwoRavens is somewhat limited at the moment (as of Spring of 2017).**

Any questions regarding the features of TwoRavens, bug reports and such, should be addressed directly to the developers of the application. The [TwoRavens GitHub repository](#) and the [TwoRavens project page](#) are good places to start.

For now, the Dataverse project will continue providing installation and integration support. We have created a new (as of Dataverse v.4.6.1) version of the installer scripts and updated this guide. We have tried to improve and simplify the installation process, particularly the difficult process of installing correct versions of the required third party R packages.

**Note that the installation process below supercedes the basic R setup described in the “Prerequisites” portion of the Installation Guide. Meaning that once completed, it installs everything needed to run TwoRavens, PLUS all the libraries and components required to ingest RData files, export as RData, and use Data Explorer.**

Please be warned:

- This process may still require some system administration skills.
- The guide below is very Linux-specific. This process has been tested on RedHat/CentOS servers only. In some ways it *may* actually be easier to get it all installed on MacOS X (because MacOS X versions of third party R packages are available pre-compiled), or even on Windows. But it hasn't been attempted, and is not supported by the Dataverse team.

In addition to the TwoRavens web application proper, several required components need to be installed and configured. This includes R, rApache and a collection of required third-party R packages. The installation steps for these components are described in the individual sections of the document below.

**Contents:**

- 0. Overview
- 1. Prerequisites
  - a. *httpd (Apache):*
  - b. *R:*
  - c. *rApache:*
  - d. *Install the build environment for R:*
- 2. *Install Extra R Packages*
- 3. *Install the TwoRavens Application*
  - a. *download and unzip the application*
  - b. *Rename the resulting directory “dataexplore” ...*
  - c. *run the installer*
  - d. *Version conflict check (preprocess.R)*
  - e. *Enable TwoRavens Button in Dataverse*
  - f. *Perform a quick test of TwoRavens functionality*
- 4. *Appendix*
  - I. *Ports configuration discussion*
  - II. *What the r-setup.sh script does:*
  - III. *What the install.pl script does:*

### 4.7.1 0. Overview

TwoRavens is itself a compact JavaScript application that **runs on the user’s browser**. These JavaScript files, and the accompanying HTML, CSS, etc. files are served by an HTTP server (Apache) as static objects.

The statistical calculations are performed by R programs that run **on the server**. `rApache` is used as the web front end for R on the server, so that the browser application can talk to R over HTTP.

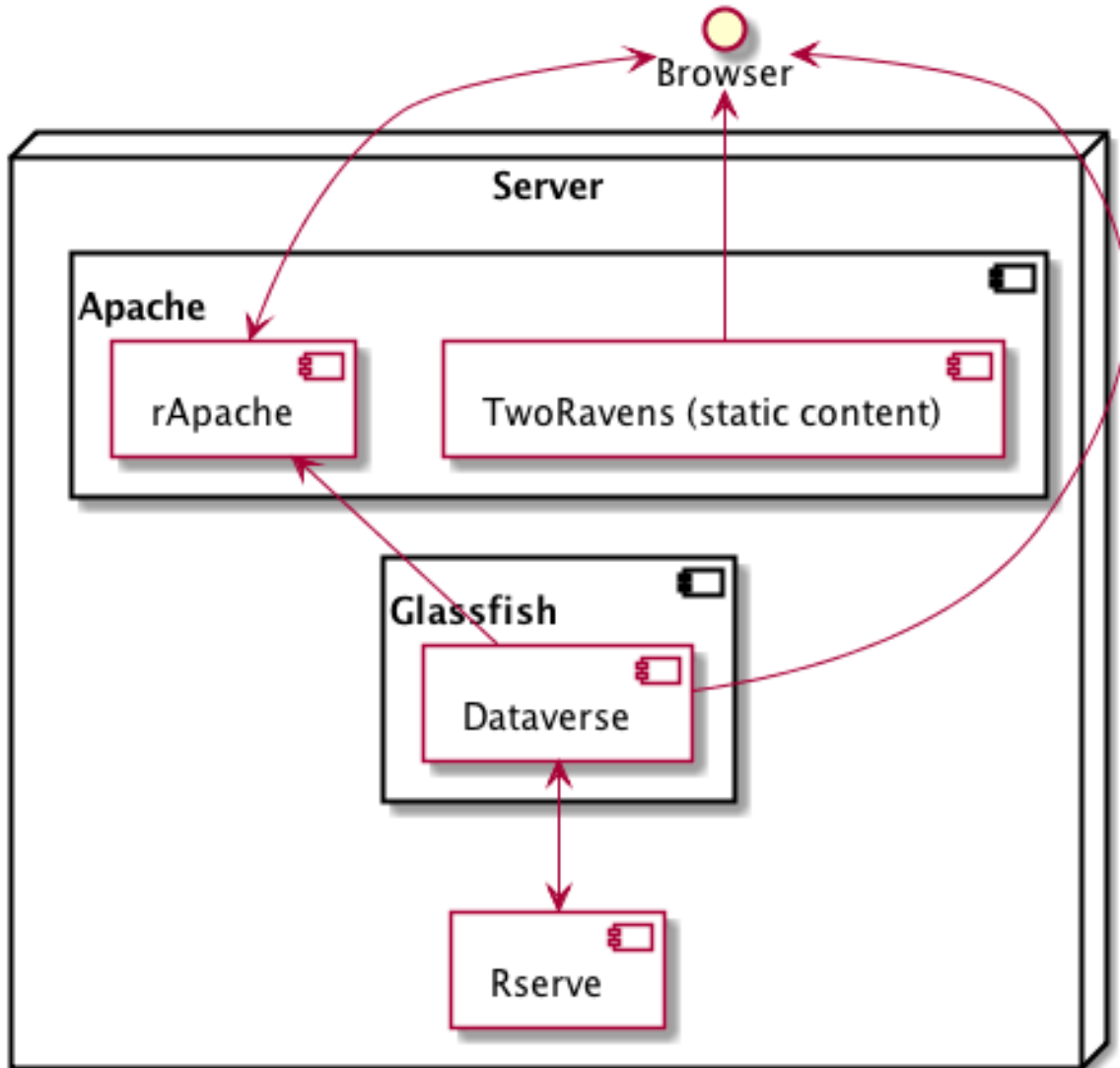
See the “Advanced Installation” section of the *Preparation* section for an example of running various components on more than one server.

TwoRavens will need to obtain some tabular-data-specific metadata from Dataverse – the DDI fragment that describes the variables and some pre-processed summary statistics for the data vectors. In order to produce the latter, the Dataverse application also needs to be able to execute some R code on the server. Instead of `rApache`, Dataverse uses `Rserve` to communicate to R. `Rserve` is installed as a “contributor” R package. It runs as a daemon process on the server, accepting network connections on a dedicated port. Dataverse project supplies an `init.d`-style startup file for the daemon. The R setup in step 2 . will set it up so that the daemon gets started automatically when the system boots.

When a user requests to run a statistical model on a data file, TwoRavens will instruct the R code on the server to download the file **directly from the Dataverse application**. Access URLs need to be configured for this to work properly (this is done by the TwoRavens installer script in step 3 .)

If you install all components on a single server and front Glassfish with Apache (see “Network Ports” under the *Configuration* section), the component and data flow diagram might look something like this:





In addition to Rserve, there are 14 more R library packages that the TwoRavens R code requires in order to run. These in turn require 30 more as their own dependencies, so a total of 45 packages must be installed. “Installed” in the context of an R package means R must download the **source code** from the [CRAN](#) code repository and compile it locally. This historically has been the trickiest, least stable part of the installation process, since the packages in question are being constantly (and independently) developed. This means that every time you attempt to install these packages, you are building from potentially different versions of the source code. An incompatibility introduced between any two of the packages can result in a failure to install. In this release we have attempted to resolve this by installing the **specific versions of the R packages that have been proven** to work together. If you have attempted to install TwoRavens in the past, and it didn’t work, please see the part of section 1.b. where we explain how to completely erase all the previously built packages.

## 4.7.2 1. Prerequisites

### a. httpd (Apache):

It’s probably installed already, but if not:

```
yum install httpd
```

This rApache configuration does not work with SELinux. Execute the following commands to disable SELinux:

```
setenforce permissive
```

```
getenforce
```

(Note: If you can get rApache to work with SELinux, we encourage you to make a pull request! Please see the [SELinux](#) section of the Developer Guide to get started.)

If you choose to serve TwoRavens and run rApache under https, a “real” signed certificate (as opposed to self-signed) is recommended.

For security reasons, directory listing needs to be disabled on the web documents folder served by Apache:

In the main Apache configuration file (`/etc/httpd/conf/httpd.conf` in the default setup), find the section that configures your web directory. For example, if the `DocumentRoot`, defined elsewhere in the file, is set to the default `"/var/www/html"`, the opening line of the section will look like this:

```
<Directory "/var/www/html">
```

Find the `Options` line in that section, and make sure that it doesn’t contain the `Indexes` statement. For example, if the options line in your configuration is

```
Options Indexes FollowSymLinks
```

change it to

```
Options FollowSymLinks
```

### b. R:

The simplest way to install R on RHEL/CentOS systems is with yum, using the EPEL repository:

```
yum install epel-release
yum install R-core R-core-devel
```

Both EPEL6 and EPEL7 currently provide R 3.5, which has been tested and appears to work well. R 3.4, offered by EPEL until also works well. We recommend using the currently available EPEL version for all the new installations. But if you already have a working R 3.4 installation from EPEL and you don’t have a specific need to upgrade, you may lock that version in place using the `yum-versionlock` yum plugin, or simply add this line to the “epel” section of `/etc/yum.repos.d/epel.repo`:

```
exclude=R-*,openblas-*,libRmath*
```

RHEL users may need to log in to their organization’s respective RHN interface, find the particular machine in question and:

- click on “Subscribed Channels: Alter Channel Subscriptions”
- enable EPEL, Server Extras, Server Optional

If you are upgrading an existing installation of TwoRavens, or if you have attempted to install it in the past and it didn’t work, **we strongly recommend reinstalling R completely**, erasing all the extra R packages that may have been already built.

Uninstall R:

```
yum erase R-core R-core-devel
```

Wipe clean any R packages that were left behind:

```
rm -rf /usr/lib64/R/library/*
rm -rf /usr/share/R/library/*
```

... then re-install R with `yum install`

### c. rApache:

We maintain the following rpms of rApache, built for the following version of RedHat/CentOS distribution:

For RHEL/CentOS 6 and R 3.4, download `rapache-1.2.6-rpm0.x86_64.rpm` and install it with:

```
yum install rapache-1.2.6-rpm0.x86_64.rpm
```

For RHEL/CentOS 6 and R 3.5, download `rapache-1.2.9_R-3.5-RH6.x86_64.rpm` and install it with:

```
yum install rapache-1.2.9_R-3.5-RH6.x86_64.rpm
```

If you are using RHEL/CentOS 7 and R 3.4, download `rapache-1.2.7-rpm0.x86_64.rpm` and install it with:

```
yum install rapache-1.2.7-rpm0.x86_64.rpm
```

If you are using RHEL/CentOS 7 in combination with R 3.5, download `rapache-1.2.9_R-3.5.x86_64.rpm` and install it with:

```
yum install rapache-1.2.9_R-3.5.x86_64.rpm
```

**Please note:** The rpms above cannot be *guaranteed* to work on your system. You may have a collection of system libraries installed on your system that will create a version conflict. If that's the case, or if you are trying to install on an operating system that's listed above, do not despair: simply build rApache from [source](#). **Make sure** to build with the R that's the same version you are planning on using.

### d. Install the build environment for R:

Once again, extra R packages will need to be built from sources. Make sure you have the standard GNU compilers installed: `gcc`, `gcc-c++` and `gcc-gfortran`.

One of the required packages needed `/bin/ed`. The R package build script needs `/usr/bin/wget`. If these are missing, the rpms can be installed with:

```
yum install ed wget
```

Depending on how your system was originally set up, you may end up needing to install some other missing rpms. We'll explain how to troubleshoot compiler errors caused by missing libraries and/or executables.

## 4.7.3 2. Install Extra R Packages

We provide a shell script (`r-setup.sh`) that will try to install all the needed packages. **Note:** the script is now part of the TwoRavens distribution (it **used to be** in the Dataverse source tree).

The script will attempt to download the packages from CRAN (or a mirror), so the system must have access to the Internet.

In order to run the script:

Download the current snapshot of the “dataverse-distribution” branch of TwoRavens from github: <https://github.com/IQSS/TwoRavens/archive/dataverse-distribution.zip>. Once again, it is important that you download the “dataverse-distribution” branch, and NOT the master distribution! Unpack the zip file, then run the script:

```
unzip dataverse-distribution.zip
cd TwoRavens-dataverse-distribution/r-setup
chmod +x r-setup.sh
./r-setup.sh
```

See the section II . of the Appendix for trouble-shooting tips.

For the Rserve package the setup script will also create a system user rserve, and install the startup script for the daemon (/etc/init.d/rserve). The script will skip this part, if this has already been done on this system (i.e., it should be safe to run it repeatedly).

Note that the setup will set the Rserve password to “rserve”. Rserve daemon runs under a non-privileged user id, and there appears to be a very limited potential for security damage through unauthorized access. It is however still a good idea **to change the password**. The password is specified in /etc/Rserv.pwd. Please see [Rserve documentation](#) for more information on password encryption and access security.

Make sure the rserve password is correctly specified in the domain.xml of your Dataverse:

```
<jvm-options>-Ddataverse.rserve.password=...</jvm-options>
```

### 4.7.4 3. Install the TwoRavens Application

#### a. download and unzip the application

(though you may have already done so, in step 2 . above - see the instructions there).

#### b. Rename the resulting directory “dataexplore” ...

...and place it in the web root directory of your apache server. We’ll assume /var/www/html/dataexplore in the examples below:

```
mv TwoRavens-dataverse-distribution /var/www/html/dataexplore
```

#### c. run the installer

A scripted, interactive installer is provided at the top level of the TwoRavens distribution.

The installer will ask you to provide the following:

Setting	default	Comment
TwoRavens directory	/var/www/html/dataexplore	File directory where TwoRavens is installed.
Apache config dir.	/etc/httpd	rApache config file for TwoRavens will be placed under conf.d/ there.
Apache web dir.	/var/www/html	
rApache/TwoRavens URL	http://{your hostname}:80	URL of the Apache server hosting TwoRavens and rApache.
Dataverse URL	http://{your hostname}:8080	URL of the Dataverse that integrates with this TwoRavens installation.

Please note the default values above. The installer assumes

- that you are running both the Dataverse and TwoRavens/rApache on the same host;
- the default ports for Apache (80) and Glassfish that is serving your Dataverse (8080);
- http (not https!) for both .

This configuration is recommended if you are simply trying out/testing Dataverse and TwoRavens. Accept all the defaults, and you should have a working installation in no time.

However, if you are planning to use this installation to actually serve data to users, you'll most likely want to run under HTTPS. Please refer to the discussion in the Appendix, I . for more information on setting it up. Configuring HTTPS takes a little extra work. But note that the TwoRavens configuration can actually end up being simpler. If you use our recommended configuration for HTTPS (described in the Appendix), both the “TwoRavens URL” and “Dataverse URL” **will be the same**: `https://{your hostname}`.

Run the installer as:

```
cd /var/www/html/dataexplore
chmod +x install.pl
./install.pl
```

Once everything is installed and configured, the installer script will print out a confirmation message with the URL of the TwoRavens application. For example:

```
The application URL is https://server.dataverse.edu/dataexplore/gui.html
```

#### d. Version conflict check (preprocess.R)

One of the R files in the TwoRavens distribution, `rook/preprocess/preprocess.R` is used by both TwoRavens and Dataverse. Dataverse application maintains its own copy of the file, `<DOMAIN DIRECTORY>/applications/dataverse-<VERSION>/WEB-INF/classes/edu/harvard/iq/dataverse/rserve/scripts/preprocess.R`. (Why not share the file from the same location? Because the two applications can potentially be installed on 2 different servers). Compare the two files. **It is important that the two copies are identical.**

**If different:**

- the **TwoRavens version wins**. Meaning, you need to copy the version supplied with this TwoRavens distribution and overwrite the Glassfish version (above); then restart Glassfish.
- unless this is a brand new Dataverse installation, it may have cached summary statistics fragments that were produced with the older version of this R code. You **must remove** all such cached files:

```
cd <DATAVERSE FILES DIRECTORY>
find . -name '*.prep' | while read file; do /bin/rm $file; done
```

*(Yes, this is a HACK! We are working on finding a better way to ensure this compatibility between TwoRavens and Dataverse!)*

#### e. Enable TwoRavens Button in Dataverse

Now that you have installed TwoRavens, you can make it available to your users by adding it an “external tool” for your Dataverse installation. (For more on external tools in general, see the [External Tools](#) section of the Admin Guide.)

First, download `twoRavens.json` as a starting point and edit `toolUrl` in that external tool manifest file to be the URL where you want TwoRavens to run. This is the URL reported by the installer script (as in the example at the end of step `c.`, above).

Once you have made your edits, make the tool available within Dataverse with the following `curl` command (assuming `twoRavens.json` is in your current working directory):

```
curl -X POST -H 'Content-type: application/json' --upload-file twoRavens.json http://localhost:8080/api/admin/externalTools
```

Once enabled, an “Explore” dropdown will appear next to ingested tabular data files a “TwoRavens” button; clicking it will redirect the user to the instance of TwoRavens, initialized with the data variables from the selected file.

### f. Perform a quick test of TwoRavens functionality

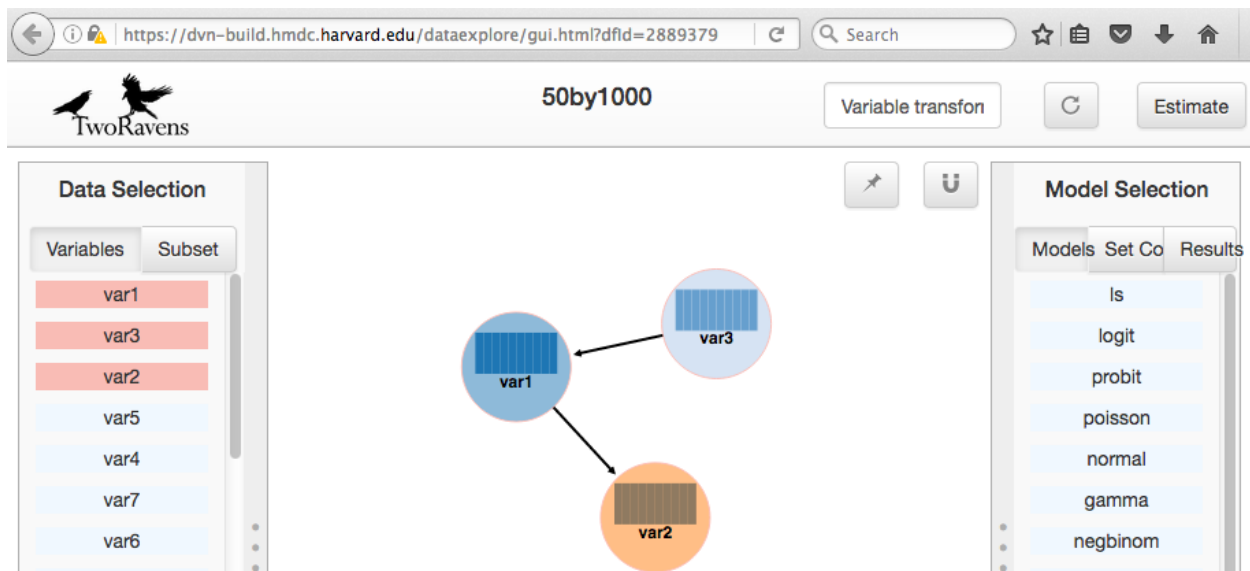
Ingest the dummy data file `50by1000.dta` (supplied in the Dataverse source tree in `dataverse/scripts/search/data/tabular`). If successfully ingested as tabular data, the file should appear on the Dataset page as follows:



If the file does NOT appear as Tabular Data - if it is shown as Stata/dta, and no tabular attributes - the numbers of Variables and Observations and the UNF - are being displayed, try to refresh the page a couple of times. If that doesn't change the view to Tabular, it likely means that something went very wrong with the tabular ingest. Consult the Glassfish server log for any error messages that may explain the failure.

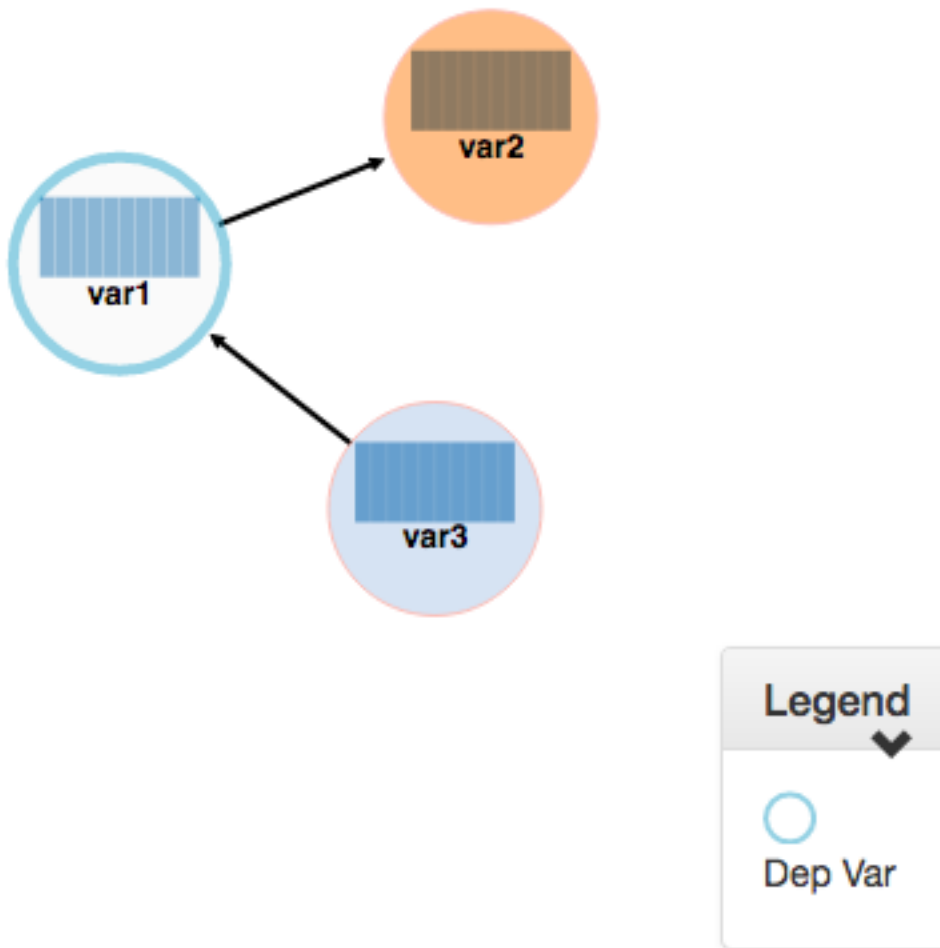
If the file is showing as Tabular Data, but the `Explore` button isn't present, double-check that the steps in `e.`, above, were correctly performed.

Otherwise, click on the `Explore` button. This will open TwoRavens in a new browser window. If the application initializes successfully, you should see the “data pebbles” representing the first 3 variables in the file:

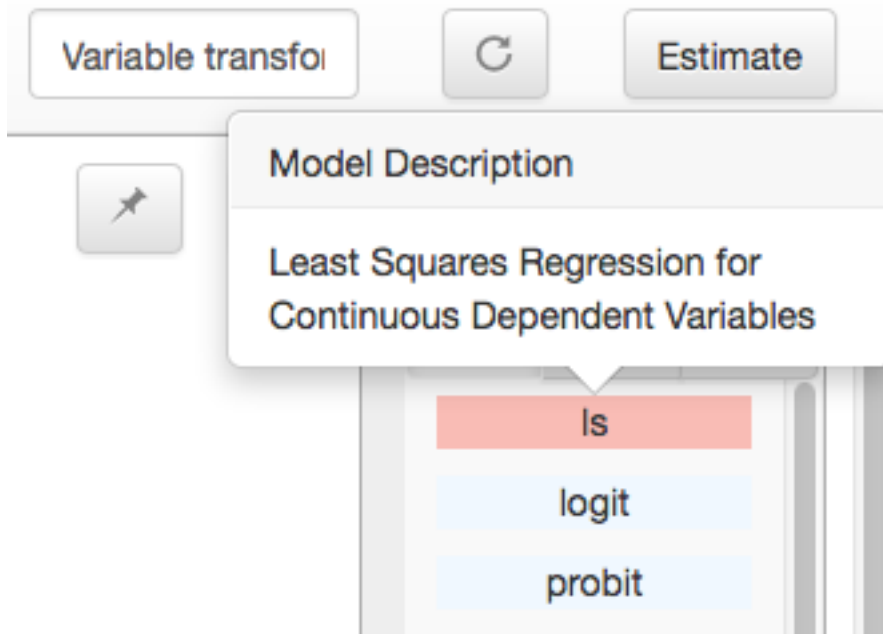


If instead TwoRavens opens with an empty view - no variables listed on the left, and/or no “data pebbles” in the middle panel, we'll provide some diagnostics tips further below.

Otherwise, mouse over `var1`, and click on `Dep Var`, selecting the variable as “dependent”:



Then select `ls` from the list of models on the right:



Then click the `Estimate` button, above. If the model is successfully executed, the results will appear in a new popup panel, with some generated graph images, as shown below:



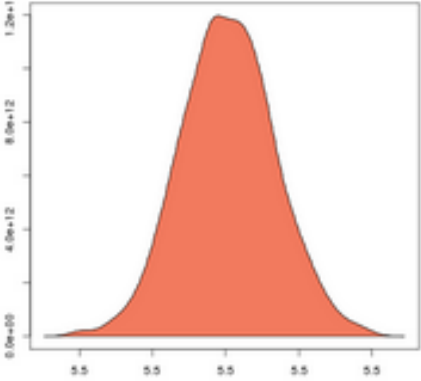
Variable transformation
↻
Estimate

### Model Selection

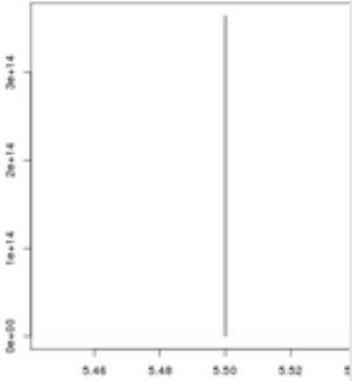
Models
Set Covar.
Results

Model1

Predicted Values: Y(X)



Expected Values: E(Y(X))



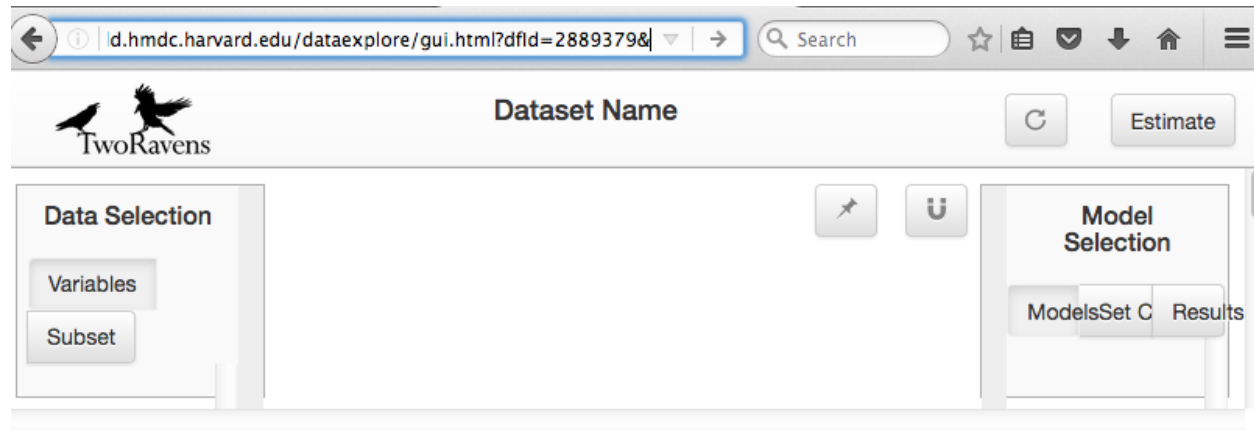
	Estimate	SE	t-value	Pr(< t )
(Intercept)	6.38e-14	2.22e-15	28.8	4.92e-133
var3	1.00	3.57e-16	2.80e+15	0.00

**Formula:** ls( var1 ~ var3 )

**Troubleshooting:**

If TwoRavens fails to initialize properly:

Symptom: instead of the “data pebbles” display shown in the second image, above, you are getting an empty view:



A very likely cause of this condition is TwoRavens not being able to obtain the metadata describing the variables from your Dataverse. Specifically, the “preprocessed summary statistics”.

To diagnose: note the value of the `dfid` URL parameter in the view above. Try to request the preprocessed fragment by going to the API end point directly:

```
<YOUR DATAVERSE URL>/api/access/datafile/<FILE ID>?format=prep
```

Where the `<FILE ID>` is the value of the `dfid` parameter from the previous view. You should get the output that looks like this:

```
{ "dataset": { "private": false }, "variables": { "var1": { "plottype": "bar", "plotvalues": { "1": 100, "2": 100, "3": 100, "4": 100, "5": 100, "6": 100, "7": 100, "8": 100, "9": 100, "10": 100 },
↪ "varnamesSumStat": "var1", "median": 5.5, "mean": 5.5, "mode": "1", "max": 10, "min": 1,
↪ "invalid": 0, "valid": 1000, "sd": 2.87371854193452, "uniques": 10, "herfindahl": 0.1,
↪ "freqmode": 100, "fewest": "1", "mid": "1", "freqfewest": "100", "freqmid": "100", "numchar":
↪ "numeric", "nature": "ordinal", "binary": "no", "interval": "discrete", "varnamesTypes":
↪ "var1", "defaultInterval": "discrete", "defaultNumchar": "numeric", "defaultNature":
↪ "ordinal", "defaultBinary": "no" }, "var3": { "plottype": "bar", "plotvalues":
...

```

If you are getting an error message instead, this is likely an Rserve connection problem. Consult the Glassfish server log for any Rserve-related “connection refused” messages. See if Rserve is running, and start it with `service rserve start`, if necessary. Check if the Rserve host name, username and password in the Glassfish configuration match the actual Rserve configuration. (this is discussed in the section 2. of the guide). Correct this, if necessary, then try again.

If you ARE getting JSON output, but the TwoRavens view is still broken:

- Look closely at the very beginning of the JSON fragment. Does it have the `{ "private": false }` entry, as shown in the example above? If not, this likely an R code version mismatch, described in section 3. d., above. Correct the problem as described there, then try again.
- If the JSON looks *exactly* as the fragment above, yet still no data pebbles - enable the JavaScript error console in the TwoRavens window, and try again. Look for any error messages; and, specifically, for any URLs that TwoRavens is failing to access. Look for the debugging entry that shows TwoRavens attempting to download the `format=prep` fragment. Does the URL have the correct host name, port and/or the protocol (http vs. https)? If not, re-run the installer, specifying the correct Dataverse URL, and try again.

Symptom: the variables view is initialized properly, but no model output appears when you click `Estimate`, with or without error messages.

- Make sure you properly selected the dependent variable (var1) and the model (ls).
- Consult the Apache error log files (`error_log` and/or `ssl_error_log`, in `/var/log/httpd`) for any error messages. Possible error condition may include: missing R packages (double-check that the R setup, in step 2, completed without errors); `selinux` (“Secure Linux”) errors related to the `rApache` shared libraries, or directory permissions (disable `Selinux`, as described in 1.a.)

## 4.7.5 4. Appendix

### I. Ports configuration discussion

By default, Glassfish will install itself on ports 8080 and 8181 (for `HTTP` and `HTTPS`, respectively). Apache will install itself on port 80 (the default port for `HTTP`). Under this configuration, your Dataverse will be accessible at `http://{your host}:8080`, and `rApache` at `http://{your host}/`. The TwoRavens installer, above, will default to these values (and assume you are running both the Dataverse and TwoRavens/`rApache` on the same host).

This configuration is the easiest to set up if you are simply trying out/testing the Dataverse and TwoRavens integration. Accept all the defaults, and you should have a working installation in no time. However, if you are planning to use this installation to actually serve data to real users, you will most likely want to run your Dataverse on a standard port; and to use `HTTPS`. It is definitely possible to configure Glassfish to serve the application under `HTTPS` on port 443. However, we **do not recommend** this setup! For at least 2 reasons: 1. Running Glassfish on port 443 will require you to **run it as root** user; which should be avoided, if possible, for reasons of security. Also, 2) installing `SSL` certificates under Glassfish is unnecessarily complicated. The alternative configuration that we recommend is to “hide” your Glassfish behind Apache. In this setup Apache serves as the `HTTPS` front running on port 443, proxying the traffic to Glassfish using `mod_proxy_ajp`; and Glassfish is running as a non-privileged user on a high port that’s not accessible from the outside. Unlike Glassfish, Apache has a mechanism for running on a privileged port (in this case, 443) as a non-privileged user. It is possible to use this configuration, and have this Apache instance serve TwoRavens and `rApache` too, all on the same server. Please see “Network Ports” under the *Configuration* section, and the *Shibboleth* section of the Installation Guide for more information and configuration instructions.

### II. What the `r-setup.sh` script does:

The script uses the list of 45 R library packages and specified package versions, supplied in `TwoRavens/r-setup/package-versions.txt` to replicate the library environment that has been proven to work on the Dataverse servers.

If any packages fail to build, the script will alert the user.

For every package, the (potentially verbose) output of the build process is saved in its own file, `RINSTALL.{PACKAGE NAME}.LOG`. So if, for example, the package Zelig fails to install, the log file `RINSTALL.Zelig.LOG` should be consulted for any error messages that may explain the reason for the failure; such as a missing library, or a missing compiler, etc. Be aware that diagnosing compiler errors will require at least some programming and/or system administration skills.

### III. What the `install.pl` script does:

The steps below are performed by the `install.pl` script. **Provided for reference only!** The instruction below could be used to configure it all by hand, if necessary, or to verify that the installer has done it correctly. Once again: **normally you would NOT need to individually perform the steps below!**

TwoRavens is distributed with a few hard-coded host and directory names. So these need to be replaced with the values specific to your system.

**In the file `/var/www/html/dataexplore/app_ddi.js` the following 3 lines need to be edited:**

1. `var production=false;`  
changed to `true`;
2. `hostname="localhost:8080";`  
changed to point to the dataverse app, from which TwoRavens will be obtaining the metadata and data files.  
(don't forget to change 8080 to the correct port number!)
3. `var rappURL = "http://0.0.0.0:8000/custom/";`  
changed to the URL of your rApache server, i.e.  
`"http(s)://<rapacheserver>:<rapacheport>/custom/";`

**In `dataexplore/rook` the following files need to be edited:**

`rookdata.R`, `rookzelig.R`, `rooksubset.R`, `rooktransform.R`, `rookselector.R`,  
`rooksource.R`

replacing *every* instance of `production<-FALSE` line with `production<-TRUE`.

(yeah, that's why we provide that installer script...)

**In `dataexplore/rook/rooksource.R` the following line:**

```
setwd("/usr/local/glassfish4/glassfish/domains/domain1/docroot/dataexplore/  
rook")
```

needs to be changed to:

```
setwd("/var/www/html/dataexplore/rook")
```

(or your `dataexplore` directory, if different from the above)

**In `dataexplore/rook/rookutils.R` the following lines need to be edited:**

```
url <- paste("https://beta.dataverse.org/custom/preprocess_dir/  
preprocessSubset_", sessionid, ".txt", sep="")
```

and

```
imageVector[[qicount]]<-paste("https://beta.dataverse.org/custom/pic_dir/",  
mysessionid, "_", mymodelcount, qicount, ".png", sep = "")
```

changing the URL to reflect the correct location of your rApache instance. make sure that the protocol (`http` vs. `https`) and the port number are correct too, not just the host name!

**Next, in order to configure rApache to serve several TwoRavens “mini-apps”,**

the installer creates the file `tworavens-rapache.conf` in the Apache's `/etc/httpd/conf.d` directory with the following configuration:

```
RSourceOnStartup "/var/www/html/dataexplore/rook/rooksource.R"  
<Location /custom/zeligapp>  
  SetHandler r-handler  
  RFileEval /var/www/html/dataexplore/rook/rookzelig.R:Rook::Server$call(zelig.app)  
</Location>  
<Location /custom/subsetapp>  
  SetHandler r-handler  
  RFileEval /var/www/html/dataexplore/rook/rooksubset.R:Rook::Server$call(subset.app)  
</Location>  
<Location /custom/transformapp>  
  SetHandler r-handler  
  RFileEval /var/www/html/dataexplore/rook/rooktransform.R:Rook::Server  
↪$call(transform.app)
```

```
</Location>
<Location /custom/dataapp>
  SetHandler r-handler
  RFileEval /var/www/html/dataexplore/rook/rookdata.R:Rook::Server$call(data.app)
</Location>
```

**The following directories are created by the installer to store various output files produced by TwoRavens:**

```
mkdir --parents /var/www/html/custom/pic_dir
mkdir --parents /var/www/html/custom/preprocess_dir
mkdir --parents /var/www/html/custom/log_dir
```

**The ownership of the TwoRavens directories is changed to user apache:**

```
chown -R apache.apache /var/www/html/custom
chown -R apache /var/www/html/dataexplore
```

**Finally, the installer restarts Apache, for all the changes to take effect:**

```
service httpd restart
```

## 4.8 Geoconnect

Geoconnect works as a middle layer, allowing geospatial data files in Dataverse to be visualized with Harvard WorldMap.

To understand the feature from the user perspective, see the *WorldMap: Geospatial Data Exploration* section of the User Guide.

As of this writing, the README at <https://github.com/IQSS/geoconnect> recommends not installing Geoconnect at this time due to an ongoing rewrite of the WorldMap code. If you are not deterred by this, read on!

To set up a Geoconnect development environment, you can follow the steps outlined in the *local\_setup.md* guide. Although those instructions are for a local development setup, they may assist in installing Geoconnect in your production environment. See also “Geoconnect” under the *Development Environment* section of the Developer Guide.

Harvard Dataverse runs Geoconnect on Heroku. To make use of Heroku, you will need a Heroku account, as well as a few other prerequisites. Follow the instructions outlined in the *heroku\_setup.md* guide. The *heroku.py* settings file may also be adapted for other environments. Please note, for the production environment, remember to set `DEBUG=False`.

See also the *Geoconnect and WorldMap* section of the Admin Guide.

## 4.9 Shibboleth

### Contents:

- *Introduction*
- *Installation*
  - *System Requirements*

- *Install Apache*
  - *Install Shibboleth*
    - \* *Enable Shibboleth Yum Repo*
    - \* *Install Shibboleth Via Yum*
- *Configure Glassfish*
  - *Apply GRIZZLY-1787 Patch*
  - *Glassfish HTTP and HTTPS ports*
  - *AJP*
  - *SSLEngine Warning Workaround*
- *Configure Apache*
  - *Enforce HTTPS*
  - *Edit Apache ssl.conf File*
- *Configure Shibboleth*
  - *shibboleth2.xml*
    - \* *Specific Identity Provider(s)*
    - \* *Identity Federation*
  - *Shibboleth Attributes*
  - *attribute-map.xml*
  - *Shibboleth and ADFS*
- *Disable or Reconfigure SELinux*
  - *Disable SELinux*
  - *Reconfigure SELinux to Accommodate Shibboleth*
    - \* *Put Type Enforcement (TE) File in misc directory*
    - \* *Navigate to misc directory*
    - \* *Run checkmodule*
    - \* *Run semodule\_package*
    - \* *Run semodule*
- *Restart Apache and Shibboleth*
- *Configure Apache and shibd to Start at Boot*
- *Verify DiscoFeed and Metadata URLs*
- *Add the Shibboleth Authentication Provider to Dataverse*
- *Exchange Metadata with Your Identity Provider*
- *Backup sp-cert.pem and sp-key.pem Files*
- *Debugging*
- *Converting Accounts*

- *Converting Local Users to Shibboleth*
- *Converting Shibboleth Users to Local*
- *Institution-Wide Shibboleth Groups*

### 4.9.1 Introduction

By configuring and enabling Shibboleth support in Dataverse, your users will be able to log in using the identity system managed by their institution (“single sign on”, or at least “single password”) rather than having to create yet another password local to your Dataverse installation. Typically, users know their login system by some sort of internal branding such as “HarvardKey” or “Touchstone” (MIT) but within the Dataverse application, the Shibboleth feature is known as “Institutional Log In” as explained to end users in the *Account Creation + Management* section of the User Guide.

Shibboleth is an implementation of the [Security Assertion Markup Language \(SAML\)](#) protocol which is similar in spirit to systems used by many webapps that allow you to log in via Google, Facebook, or Twitter.

Shibboleth can be compared and contrasted with OAuth2, which you can read about in the *OAuth Login: ORCID, GitHub, Google* section.

### 4.9.2 Installation

We assume you’ve already gone through a basic installation as described in the *Installation* section and that you’ve paid particular attention to the “Auth Modes: Local vs. Remote vs. Both” explanation in the *Configuration* section. You’re going to give Shibboleth a whirl. Let’s get started.

#### System Requirements

Support for Shibboleth in Dataverse is built on the popular “[mod\\_shib](#)” Apache module, “[shibd](#)” daemon, and the [Embedded Discovery Service \(EDS\)](#) Javascript library, all of which are distributed by the [Shibboleth Consortium](#). EDS is bundled with Dataverse, but `mod_shib` and `shibd` must be installed and configured per below.

Only Red Hat Enterprise Linux (RHEL) and derivatives such as CentOS have been tested (x86\_64 versions) by the Dataverse team. See <https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPLinuxInstall> for details and note that (according to that page) as of this writing Ubuntu and Debian are not officially supported by the Shibboleth project.

#### Install Apache

We will be “fronting” Glassfish with Apache so that we can make use of the `mod_shib` Apache module. We will also make use of the `mod_proxy_ajp` module built in to Apache.

We include the `mod_ssl` package to enforce HTTPS per below.

```
yum install httpd mod_ssl
```

#### Install Shibboleth

Installing Shibboleth will give us both the `shibd` service and the `mod_shib` Apache module.

## Enable Shibboleth Yum Repo

This yum repo is recommended at <https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPLinuxRPMInstall>  
cd /etc/yum.repos.d

If you are running el7 (RHEL/CentOS 7):

```
wget http://download.opensuse.org/repositories/security:/shibboleth/CentOS_7/
security:shibboleth.repo
```

If you are running el6 (RHEL/CentOS 6):

```
wget http://download.opensuse.org/repositories/security:/shibboleth/
CentOS_CentOS-6/security:shibboleth.repo
```

## Install Shibboleth Via Yum

```
yum install shibboleth
```

## 4.9.3 Configure Glassfish

### Apply GRIZZLY-1787 Patch

In order for the Dataverse “download as zip” feature to work well with large files without causing `OutOfMemoryError` problems on Glassfish 4.1 when fronted with Apache, you should stop Glassfish, with `./asadmin stop-domain domain1`, make a backup of `glassfish4/glassfish/modules/glassfish-grizzly-extra-all.jar`, replace it with a patched version of `glassfish-grizzly-extra-all.jar` downloaded from [here](#) (the md5 is in the README), and start Glassfish again with `./asadmin start-domain domain1`.

For more background on the patch, please see <https://java.net/jira/browse/GRIZZLY-1787> and <https://github.com/IQSS/dataverse/issues/2180> and <https://github.com/Payara/Payara/issues/350>

This problem has been reported to Glassfish at <https://java.net/projects/glassfish/lists/users/archive/2015-07/message/1> and while Glassfish 4.1.1 includes a new enough version of Grizzly to fix the bug, other complicating factors prevent its adoption (look for “Glassfish 4.1.1” in the *Prerequisites* section for details on why it is not recommended).

### Glassfish HTTP and HTTPS ports

Apache will be listening on ports 80 and 443 so we need to make sure Glassfish isn’t using them. If you’ve been changing the default ports used by Glassfish per the *Configuration* section, revert the Glassfish HTTP service to listen on 8080, the default port:

```
./asadmin set server-config.network-config.network-listeners.network-listener.
http-listener-1.port=8080
```

Likewise, if necessary, revert the Glassfish HTTPS service to listen on port 8181:

```
./asadmin set server-config.network-config.network-listeners.network-listener.
http-listener-2.port=8181
```



## AJP

A `jk-connector` network listener should have already been set up when you ran the installer mentioned in the *Installation* section, but for reference, here is the command that is used:

```
./asadmin create-network-listener --protocol http-listener-1 --listenerport
8009 --jkenabled true jk-connector
```

You can verify this with `./asadmin list-network-listeners`.

This enables the [AJP protocol](#) used in Apache configuration files below.

## SSLEngine Warning Workaround

When fronting Glassfish with Apache and using the `jk-connector` (AJP, `mod_proxy_ajp`), in your Glassfish `server.log` you can expect to see “WARNING ... `org.glassfish.grizzly.http.server.util.RequestUtils ... jk-connector ... Unable to populate SSL attributes java.lang.IllegalStateException: SSLEngine is null`”.

To hide these warnings, run `./asadmin set-log-levels org.glassfish.grizzly.http.server.util.RequestUtils=SEVERE` so that the `WARNING` level is hidden as recommended at <https://java.net/jira/browse/GLASSFISH-20694> and <https://github.com/IQSS/dataverse/issues/643#issuecomment-49654847>

## 4.9.4 Configure Apache

### Enforce HTTPS

To prevent attacks such as [FireSheep](#), `HTTPS` should be enforced. <https://wiki.apache.org/httpd/RewriteHTTPToHTTPS> provides a good method. You **could** copy and paste that those “rewrite rule” lines into Apache’s main config file at `/etc/httpd/conf/httpd.conf` but using Apache’s “virtual hosts” feature is recommended so that you can leave the main configuration file alone and drop a host-specific file into place.

Below is an example of how “rewrite rule” lines look within a `VirtualHost` block. Download a sample file, edit it to substitute your own hostname under `ServerName`, and place it at `/etc/httpd/conf.d/dataverse.example.edu.conf` or a filename that matches your hostname. The file must be in `/etc/httpd/conf.d` and must end in “.conf” to be included in Apache’s configuration.

```
<VirtualHost *:80>

ServerName dataverse.example.edu

# From https://wiki.apache.org/httpd/RewriteHTTPToHTTPS

RewriteEngine On
# This will enable the Rewrite capabilities

RewriteCond %{HTTPS} !=on
# This checks to make sure the connection is not already HTTPS

RewriteRule ^/?(.*) https://%{SERVER_NAME}/$1 [R,L]
# This rule will redirect users from their original location, to the same location,
↳but using HTTPS.
# i.e. http://www.example.com/foo/ to https://www.example.com/foo/
# The leading slash is made optional so that this will work either in httpd.conf
# or .htaccess context

</VirtualHost>
```

## Edit Apache ssl.conf File

/etc/httpd/conf.d/ssl.conf should be edited to contain the FQDN of your hostname like this: `ServerName dataverse.example.edu:443` (substituting your hostname).

Near the bottom of /etc/httpd/conf.d/ssl.conf but before the closing `</VirtualHost>` directive, add the following:

```
# don't pass paths used by rApache and TwoRavens to Glassfish
ProxyPassMatch ^/RApacheInfo$ !
ProxyPassMatch ^/custom !
ProxyPassMatch ^/dataexplore !
# don't pass paths used by Shibboleth to Glassfish
ProxyPassMatch ^/Shibboleth.sso !
ProxyPassMatch ^/shibboleth-ds !
# pass everything else to Glassfish
ProxyPass / ajp://localhost:8009/

<Location /shib.xhtml>
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  require valid-user
</Location>
```

You can download a sample `ssl.conf` file to compare it against the file you edited.

Note that `/etc/httpd/conf.d/shib.conf` and `/etc/httpd/conf.d/shibboleth-ds.conf` are expected to be present from installing Shibboleth via yum.

You may wish to also add a timeout directive to the ProxyPass line within `ssl.conf`. This is especially useful for larger file uploads as apache may prematurely kill the connection before the upload is processed.

e.g. `ProxyPass / ajp://localhost:8009/ timeout=600` defines a timeout of 600 seconds.

Try to strike a balance with the timeout setting. Again a timeout too low will impact file uploads. A timeout too high may cause additional stress on the server as it will have to service idle clients for a longer period of time.

## 4.9.5 Configure Shibboleth

### shibboleth2.xml

/etc/shibboleth/shibboleth2.xml should look something like the sample `shibboleth2.xml` file below, but you must substitute your hostname in the `entityID` value. If your starting point is a `shibboleth2.xml` file provided by someone else, you must ensure that `attributePrefix="AJP_"` is added under `ApplicationDefaults` per the [Shibboleth wiki](#). Without the `AJP_` configuration in place, the required *Shibboleth Attributes* will be null and users will be unable to log in.

```
<!--
This is an example shibboleth2.xml generated originally by http://testshib.org
and tweaked for Dataverse. See also:

- attribute-map.xml
- dataverse-idp-metadata.xml

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPConfiguration
-->

<SPConfig xmlns="urn:mace:shibboleth:3.0:native:sp:config" xmlns:md=
↳"urn:oasis:names:tc:SAML:2.0:metadata"
```

```

xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
clockSkew="1800">

<!-- FIXME: change the entityID to your hostname. -->
<ApplicationDefaults entityID="https://dataverse.example.edu/sp"
  REMOTE_USER="eppn" attributePrefix="AJP_">

  <!-- You should use secure cookies if at all possible. See cookieProps in
↳ this Wiki article. -->
  <!-- https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPSessions -->
  <Sessions lifetime="28800" timeout="3600" checkAddress="false" relayState=
↳ "ss:mem" handlerSSL="false">

    <SSO>
      SAML2 SAML1
    </SSO>

    <!-- SAML and local-only logout. -->
    <!-- https://wiki.shibboleth.net/confluence/display/SHIB2/
↳ NativeSPServiceLogout -->
    <Logout>SAML2 Local</Logout>

    <!--
↳ information. Try them out!
    Attribute values received by the SP through SAML will be visible at:
    http://dataverse.example.edu/Shibboleth.sso/Session
    -->

    <!-- Extension service that generates "approximate" metadata based on SP
↳ configuration. -->
    <Handler type="MetadataGenerator" Location="/Metadata" signing="false"/>

    <!-- Status reporting service. -->
    <Handler type="Status" Location="/Status" acl="127.0.0.1"/>

    <!-- Session diagnostic service. -->
    <!-- showAttributeValues must be set to true to see attributes at /
↳ Shibboleth.sso/Session . -->
    <Handler type="Session" Location="/Session" showAttributeValues="true"/>

    <!-- JSON feed of discovery information. -->
    <Handler type="DiscoveryFeed" Location="/DiscoFeed"/>

  </Sessions>

  <!-- Error pages to display to yourself if something goes horribly wrong. -->
  <Errors supportContact="root@localhost" logoLocation="/shibboleth-sp/logo.jpg"
    styleSheet="/shibboleth-sp/main.css"/>

  <!-- Loads and trusts a metadata file that describes only the Testshib IdP
↳ and how to communicate with it. -->
  <!-- IdPs we want allow go in /etc/shibboleth/dataverse-idp-metadata.xml -->
  <MetadataProvider type="XML" path="dataverse-idp-metadata.xml"
↳ backingFilePath="local-idp-metadata.xml" legacyOrgNames="true" reloadInterval="7200
↳ "/>

  <!-- Uncomment to enable all the Research & Scholarship IdPs from InCommon -->
  <!--

```

```

    <MetadataProvider type="XML" url="http://md.incommon.org/InCommon/InCommon-
↪ metadata.xml" backingFilePath="InCommon-metadata.xml" maxRefreshDelay="3600">
        <DiscoveryFilter type="Whitelist" matcher="EntityAttributes">
            <saml:Attribute
                Name="http://macedir.org/entity-category-support"
                NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
                <saml:AttributeValue>http://id.incommon.org/category/research-and-
↪ scholarship</saml:AttributeValue>
            </saml:Attribute>
            <saml:Attribute
                Name="http://macedir.org/entity-category-support"
                NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
                <saml:AttributeValue>http://refeds.org/category/research-and-
↪ scholarship</saml:AttributeValue>
            </saml:Attribute>
        </DiscoveryFilter>
    </MetadataProvider>
-->

<!-- Attribute and trust options you shouldn't need to change. -->
<AttributeExtractor type="XML" validate="true" path="attribute-map.xml"/>
<AttributeResolver type="Query" subjectMatch="true"/>
<AttributeFilter type="XML" validate="true" path="attribute-policy.xml"/>

<!-- Your SP generated these credentials. They're used to talk to IdP's. -->
<CredentialResolver type="File" key="sp-key.pem" certificate="sp-cert.pem"/>

</ApplicationDefaults>

<!-- Security policies you shouldn't change unless you know what you're doing. -->
<SecurityPolicyProvider type="XML" validate="true" path="security-policy.xml"/>

<!-- Low-level configuration about protocols and bindings available for use. -->
<ProtocolProvider type="XML" validate="true" reloadChanges="false" path=
↪ "protocols.xml"/>
</SPConfig>

```

## Specific Identity Provider(s)

When configuring the `MetadataProvider` section of `shibboleth2.xml` you should consider if your users will all come from the same Identity Provider (IdP) or not.

Most Dataverse installations will probably only want to authenticate users via Shibboleth using their home institution's Identity Provider (IdP). The configuration above in `shibboleth2.xml` looks for the metadata for the Identity Providers (IdPs) in a file at `/etc/shibboleth/dataverse-idp-metadata.xml`. You can download a sample `dataverse-idp-metadata.xml` file and that includes the TestShib IdP from <http://testshib.org> but you will want to edit this file to include the metadata from the Identity Provider(s) you care about. The identity people at your institution will be able to provide you with this metadata and they will very likely ask for a list of attributes that Dataverse requires, which are listed at *Shibboleth Attributes*.

## Identity Federation

Rather than or in addition to specifying individual Identity Provider(s) you may wish to broaden the number of users who can log into your Dataverse installation by registering your Dataverse installation as a Service Provider (SP)

within an identity federation. For example, in the United States, users from the many institutions registered with the “InCommon” identity federation that release the “Research & Scholarship Attribute Bundle” will be able to log into your Dataverse installation if you register it as an InCommon Service Provider that is part of the Research & Scholarship (R&S) category.

The details of how to register with an identity federation are out of scope for this document, but a good starting point may be this list of identity federations across the world: <http://www.protectnetwork.org/support/faq/identity-federations>

One of the benefits of using `shibd` is that it can be configured to periodically poll your identity federation for updates as new Identity Providers (IdPs) join the federation you’ve registered with. For the InCommon federation, the following page describes how to download and verify signed InCommon metadata every hour: <https://spaces.internet2.edu/display/InCFederation/Shibboleth+Metadata+Config#ShibbolethMetadataConfig-ConfiguretheShibbolethSP> . You can also see an example of this as `maxRefreshDelay="3600"` in the commented out section of the `shibboleth2.xml` file above.

Once you’ve joined a federation the list of IdPs in the dropdown can be quite long! If you’re curious how many are in the list you could try something like this: `curl https://dataverse.example.edu/Shibboleth.sso/DiscoFeed | jq '.[].entityID' | wc -l`

## Shibboleth Attributes

The following attributes are required for a successful Shibboleth login:

- Shib-Identity-Provider
- eppn
- givenName
- sn
- email

See also <https://www.incommon.org/federation/attributesummary.html> and <https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAAttributeAccess>

## attribute-map.xml

By default, some attributes `/etc/shibboleth/attribute-map.xml` are commented out. Edit the file to enable them so that all the require attributes come through. You can download a sample `attribute-map.xml` file.

## Shibboleth and ADFS

With appropriate configuration, Dataverse and Shibboleth can make use of “single sign on” using Active Directory. This requires configuring `shibd` and `httpd` to load appropriate libraries, and insuring that the attribute mapping matches those provided. Example configuration files for `shibboleth2.xml` and `attribute-map.xml` may be helpful. Note that your ADFS server hostname goes in the file referenced under “MetadataProvider” in your `shibboleth2.xml` file.

### 4.9.6 Disable or Reconfigure SELinux

SELinux is set to “enforcing” by default on RHEL/CentOS, but unfortunately Shibboleth does not “just work” with SELinux. You have two options. You can disable SELinux or you can reconfigure SELinux to accommodate Shibboleth.

## Disable SELinux

The first and easiest option is to set `SELINUX=permissive` in `/etc/selinux/config` and run `setenforce permissive` or otherwise disable SELinux to get Shibboleth to work. This is apparently what the Shibboleth project expects because their wiki page at <https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPSELinux> says, “At the present time, we do not support the SP in conjunction with SELinux, and at minimum we know that communication between the `mod_shib` and `shibd` components will fail if it’s enabled. Other problems may also occur.”

## Reconfigure SELinux to Accommodate Shibboleth

The second (more involved) option is to use the `checkmodule`, `semodule_package`, and `semodule` tools to apply a local policy to make Shibboleth work with SELinux. Let’s get started.

### Put Type Enforcement (TE) File in misc directory

Copy and paste or download the `shibboleth.te` Type Enforcement (TE) file below and put it at `/etc/selinux/targeted/src/policy/domains/misc/shibboleth.te`.

```
module shibboleth 1.0;

require {
    class file {open read};
    class sock_file write;
    class unix_stream_socket connectto;
    type httpd_t;
    type initrc_t;
    type var_run_t;
    type var_t;
}

allow httpd_t initrc_t:unix_stream_socket connectto;
allow httpd_t var_run_t:sock_file write;
allow httpd_t var_t:file {open read};
```

(If you would like to know where the `shibboleth.te` came from and how to hack on it, please see the *SELinux* section of the Developer Guide. Pull requests are welcome!)

### Navigate to misc directory

```
cd /etc/selinux/targeted/src/policy/domains/misc
```

### Run checkmodule

```
checkmodule -M -m -o shibboleth.mod shibboleth.te
```

### Run semodule\_package

```
semodule_package -o shibboleth.pp -m shibboleth.mod
```

Silent is golden. No output is expected.

## Run semodule

```
semodule -i shibboleth.pp
```

Silent is golden. No output is expected. This will place a file in `/etc/selinux/targeted/modules/active/modules/shibboleth.pp` and include “shibboleth” in the output of `semodule -l`. See the `semodule` man page if you ever want to remove or disable the module you just added.

Congrats! You’ve made the creator of <http://stopdisablinglinux.com> proud. :)

### 4.9.7 Restart Apache and Shibboleth

After configuration is complete:

```
service shibd restart
service httpd restart
```

### 4.9.8 Configure Apache and shibd to Start at Boot

```
chkconfig httpd on
chkconfig shibd on
```

### 4.9.9 Verify DiscoFeed and Metadata URLs

As a sanity check, visit the following URLs (substituting your hostname) to make sure you see JSON and XML:

- <https://dataverse.example.edu/Shibboleth.sso/DiscoFeed>
- <https://dataverse.example.edu/Shibboleth.sso/Metadata>

The JSON in `DiscoFeed` comes from the list of IdPs you configured in the `MetadataProvider` section of `shibboleth2.xml` and will form a dropdown list on the Login Page.

### 4.9.10 Add the Shibboleth Authentication Provider to Dataverse

Now that you’ve configured Glassfish, Apache, and `shibd`, you are ready to turn your attention back to Dataverse to enable Shibboleth as an “authentication provider.” You will be using `curl` to POST the following JSON file to the `authenticationProviders` endpoint of the *Native API*.

```
{
  "id": "shib",
  "factoryAlias": "shib",
  "enabled": true
}
```

```
curl -X POST -H 'Content-type: application/json' --upload-file
shibAuthProvider.json http://localhost:8080/api/admin/authenticationProviders
```

Now that you’ve added the Shibboleth authentication provider to Dataverse, as described in the *Account Creation + Management* section of the User Guide, you should see a new “Your Institution” button under “Other Log In Options” on the Log In page. After clicking “Your Institution”, you should see the institutions you configured in `/etc/shibboleth/shibboleth2.xml` above. If not, double check the content of the `DiscoFeed` URL above. If

you don't see the "Your Institution" button, confirm that the the "shib" authentication provider has been added by listing all the authentication providers Dataverse knows about:

```
curl http://localhost:8080/api/admin/authenticationProviders
```

Once you have confirmed that the Dataverse web interface is listing the institutions you expect, you'll want to temporarily remove the Shibboleth authentication provider you just added because users won't be able to log in via their institution until you have exchanged metadata with one or more Identity Providers (IdPs), which is described below. As explained in the section of the *Native API* of the API Guide, you can delete an authentication provider by passing its id:

```
curl -X DELETE http://localhost:8080/api/admin/authenticationProviders/shib
```

Before contacting your actual Identity Provider, we recommend testing first with the "TestShib" Identity Provider (IdP) to ensure that you have configured everything correctly.

### 4.9.11 Exchange Metadata with Your Identity Provider

<http://testshib.org> (TestShib) is a fantastic resource for testing Shibboleth configurations. Depending on your relationship with your identity people you may want to avoid bothering them until you have tested your Dataverse configuration with the TestShib Identity Provider (IdP). This process is explained below.

If you've temporarily configured your `MetadataProvider` to use the TestShib Identity Provider (IdP) as outlined above, you can download your metadata like this (substituting your hostname in both places):

```
curl https://dataverse.example.edu/Shibboleth.sso/Metadata > dataverse.  
example.edu
```

Then upload your metadata to <http://testshib.org/register.html>

Then try to log in to Dataverse using the TestShib IdP. After logging in, you can visit the <https://dataverse.example.edu/Shibboleth.sso/Session> (substituting your hostname) to troubleshoot which attributes are being received. You should see something like the following:

```
Miscellaneous  
Session Expiration (barring inactivity): 479 minute(s)  
Client Address: 65.112.10.82  
SSO Protocol: urn:oasis:names:tc:SAML:2.0:protocol  
Identity Provider: https://idp.testshib.org/idp/shibboleth  
Authentication Time: 2016-03-08T13:45:10.922Z  
Authentication Context Class: urn:oasis:names:tc:SAML:2.  
↪0:ac:classes>PasswordProtectedTransport  
Authentication Context Decl: (none)  
  
Attributes  
affiliation: Member@testshib.org;Staff@testshib.org  
cn: Me Myself And I  
entitlement: urn:mace:dir:entitlement:common-lib-terms  
eppn: myself@testshib.org  
givenName: Me Myself  
persistent-id: https://idp.testshib.org/idp/shibboleth!https://dataverse.example.edu/  
↪sp!RuyCiLvUcgmKqyh/rOQPh+wyR7s=  
sn: And I  
telephoneNumber: 555-5555  
uid: myself  
unscoped-affiliation: Member;Staff
```

(As of this writing the TestShib IdP does not send the "mail" attribute, a required attribute, but for testing purposes, Dataverse compensates for this for the TestShib IdP and permits login anyway.)



When you are done testing, you can delete the TestShib users you created like this (after you have deleted any data and permissions associated with the users):

```
curl -X DELETE http://localhost:8080/api/admin/authenticatedUsers/myself
```

(Of course, you are also welcome to do a fresh reinstall per the *Installation* section.)

If your Dataverse installation is working with TestShib it **should** work with your institution's Identity Provider (IdP). Next, you should:

- Send your identity people your metadata file above (or a link to download it themselves). From their perspective you are a Service Provider (SP).
- Ask your identity people to send you the metadata for the Identity Provider (IdP) they operate. See the section above on `shibboleth2.xml` and `MetadataProvider` for what to do with the IdP metadata. Restart `shibd` and `httpd` as necessary.
- Re-add Shibboleth as an authentication provider to Dataverse as described above.
- Test login to Dataverse via your institution's Identity Provider (IdP).

#### 4.9.12 Backup `sp-cert.pem` and `sp-key.pem` Files

Especially if you have gotten authentication working with your institution's Identity Provider (IdP), now is the time to make sure you have backups.

The installation and configuration of Shibboleth will result in the following cert and key files being created and it's important to back them up. The cert is in the metadata you shared with your IdP:

- `/etc/shibboleth/sp-cert.pem`
- `/etc/shibboleth/sp-key.pem`

If you have more than one Glassfish server, you should use the same `sp-cert.pem` and `sp-key.pem` files on all of them. If these files are compromised and you need to regenerate them, you can `cd /etc/shibboleth` and run `keygen.sh` like this (substituting you own hostname):

```
./keygen.sh -f -u shibd -g shibd -h dataverse.example.edu -e https://  
dataverse.example.edu/sp
```

#### 4.9.13 Debugging

The *Troubleshooting* section of the Admin Guide explains how to increase Glassfish logging levels. The relevant classes and packages are:

- `edu.harvard.iq.dataverse.Shib`
- `edu.harvard.iq.dataverse.authorization.providers.shib`
- `edu.harvard.iq.dataverse.authorization.groups.impl.shib`

#### 4.9.14 Converting Accounts

As explained in the *Account Creation + Management* section of the User Guide, users can convert from one login option to another.

## Converting Local Users to Shibboleth

If you are running in “remote and local” mode and have existing local users that you’d like to convert to Shibboleth users, give them the following steps to follow, which are also explained in the *Account Creation + Management* section of the User Guide:

- Log in with your local account to make sure you know your password, which will be needed for the account conversion process.
- Log out of your local account.
- Log in with your Shibboleth account.
- If the email address associated with your local account matches the email address asserted by the Identity Provider (IdP), you will be prompted for the password of your local account and asked to confirm the conversion of your account. You’re done! Browse around to ensure you see all the data you expect to see. Permissions have been preserved.
- If the email address asserted by the Identity Provider (IdP) does not match the email address of any local user, you will be prompted to create a new account. If you were expecting account conversion, you should decline creating a new Shibboleth account, log back in to your local account, and let Support know the email on file for your local account. Support may ask you to change your email address for your local account to the one that is being asserted by the Identity Provider. Someone with access to the Glassfish logs will see this email address there.

## Converting Shibboleth Users to Local

Whereas users convert their own accounts from local to Shibboleth as described above, conversion in the opposite direction is performed by a sysadmin. A common scenario may be as follows:

- A user emails Support saying, “I left the university (or wherever) and can’t log in to Dataverse anymore. What should I do?”
- Support replies asking the user for a new email address (Gmail, new institution email, etc.) to associate with their Dataverse account.
- The user replies with a new email address to associate with their Dataverse account.
- Support runs the curl command below, supplying the database id of the user to convert and the new email address and notes the username returned.
- Support emails the user and indicates that that they should use the password reset feature to set a new password and to make sure to take note of their username under Account Information (or the password reset confirmation email) since the user never had a username before.
- The user resets password and is able to log in with their local account. All permissions have been preserved with the exception of any permissions assigned to an institution-wide Shibboleth group to which the user formerly belonged.

In the example below, the user has indicated that the new email address they’d like to have associated with their account is “former.shib.user@mailinator.com” and their user id from the `authenticateduser` database table is “2”. The API token must belong to a superuser (probably the sysadmin executing the command). Note that the old version of this call, `convertShibToBuiltIn`, is deprecated and will be deleted in a future release.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT -d "former.shib.user@mailinator.com" http://localhost:8080/api/admin/authenticatedUsers/id/2/convertRemoteToBuiltIn
```

Rather than looking up the user’s id in the `authenticateduser` database table, you can issue this command to get a listing of all users:

```
curl -H "X-Dataverse-key: $API_TOKEN" http://localhost:8080/api/admin/
authenticatedUsers
```

Per above, you now need to tell the user to use the password reset feature to set a password for their local account.

### 4.9.15 Institution-Wide Shibboleth Groups

Dataverse allows you to optionally define “institution-wide Shibboleth groups” based on the the entityID of the Identity Provider (IdP) used to authenticate. For example, an “institution-wide Shibboleth group” with `https://idp.testshib.org/idp/shibboleth` as the IdP would include everyone who logs in via the TestShib IdP mentioned above.

To create an institution-wide Shibboleth groups, create a JSON file as below and issue this curl command: `curl http://localhost:8080/api/admin/groups/shib -X POST -H 'Content-type:application/json' --upload-file shibGroupTestShib.json`

```
{
  "name": "All testshib.org Shibboleth Users",
  "attribute": "Shib-Identity-Provider",
  "pattern": "https://idp.testshib.org/idp/shibboleth"
}
```

Institution-wide Shibboleth groups are based on the “Shib-Identity-Provider” SAML attribute asserted at runtime after successful authentication with the Identity Provider (IdP) and held within the browser session rather than being persisted in the database for any length of time. It is for this reason that roles based on these groups, such as the ability to create a dataset, are not honored by non-browser interactions, such as through the SWORD API.

To list institution-wide Shibboleth groups: `curl http://localhost:8080/api/admin/groups/shib`

To delete an institution-wide Shibboleth group (assuming id 1): `curl -X DELETE http://localhost:8080/api/admin/groups/shib/1`

Support for arbitrary attributes beyond “Shib-Identity-Provider” such as “eduPersonScopedAffiliation”, etc. is being tracked at <https://github.com/IQSS/dataverse/issues/1515>

## 4.10 OAuth Login: ORCID, GitHub, Google

### Contents:

- *Introduction*
- *Setup*
  - *Identity Provider Side*
    - \* *Obtain Client ID and Client Secret*
  - *Dataverse Side*
    - \* *ORCID Sandbox*
- *Converting Local Users to OAuth*
- *Converting OAuth Users to Local*

## 4.10.1 Introduction

As explained under “Auth Modes” in the *Configuration* section, OAuth2 is one of the ways that you can have end users log in to Dataverse.

OAuth2 is an authentication protocol that allows systems to share user data, while letting the users control what data is being shared. When you see buttons stating “login with Google” or “login through Facebook”, OAuth2 is probably involved. For the purposes of this section, we will shorten “OAuth2” to just “OAuth.” OAuth can be compared and contrasted with *Shibboleth*.

Dataverse supports three OAuth providers: ORCID, GitHub, and Google.

## 4.10.2 Setup

Setting up an OAuth identity provider to work with Dataverse requires setup in two places: the provider, and the Dataverse installation.

### Identity Provider Side

#### Obtain Client ID and Client Secret

Before OAuth providers will release information about their users (first name, last name, etc.) to your Dataverse installation, you must request a “Client ID” and “Client Secret” from them. In the case of GitHub and Google, this is as simple as clicking a few buttons and there is no cost associated with using their authentication service. ORCID, on the other hand, does not have an automated system for requesting these credentials, and it is not free to use the ORCID authentication service.

URLs to help you request a Client ID and Client Secret from the providers supported by Dataverse are provided below. For all of these providers, it’s a good idea to request the Client ID and Client secret using a generic account, perhaps the one that’s associated with the `:SystemEmail` you’ve configured for Dataverse, rather than your own personal ORCID, GitHub, or Google account:

- ORCID: <https://orcid.org/content/register-client-application-production-trusted-party>
- GitHub: <https://github.com/settings/applications/new> via <https://developer.github.com/v3/oauth/>
- Google: <https://console.developers.google.com/projectselector/apis/credentials> via <https://developers.google.com/identity/protocols/OAuth2WebServer> (pick “OAuth client ID”)

Each of these providers will require the following information from you:

- Basic information about your Dataverse installation such as a name, description, URL, logo, privacy policy, etc.
- OAuth2 Redirect URI (ORCID) or Authorization Callback URL (GitHub) or Authorized Redirect URIs (Google): This is the URL on the Dataverse side to which the user will be sent after successfully authenticating with the identity provider. This should be the advertised URL of your Dataverse installation (the protocol, fully qualified domain name, and optional port configured via the `dataverse.siteUrl` JVM option mentioned in the *Configuration* section) appended with `/oauth2/callback.xhtml` such as `https://dataverse.example.edu/oauth2/callback.xhtml`.

When you are finished you should have a Client ID and Client Secret from the provider. Keep them safe and secret.

### Dataverse Side

As explained under “Auth Modes” in the *Configuration* section, available authentication providers are stored in the `authenticationproviderrow` database table and can be listed with this command:

```
curl http://localhost:8080/api/admin/authenticationProviders
```

We will POST a JSON file containing the Client ID and Client Secret to this authenticationProviders API endpoint to add another authentication provider. As a starting point, you'll want to download the JSON template file matching the provider you're setting up:

- orcid.json
- github.json
- google.json

Here's how the JSON template for GitHub looks, for example:

```
{
  "id": "github",
  "factoryAlias": "oauth2",
  "title": "GitHub",
  "subtitle": "",
  "factoryData": "type: github | userEndpoint: NONE | clientId: FIXME | ↵
↵clientSecret: FIXME",
  "enabled": true
}
```

Edit the JSON template and replace the two “FIXME” values with the Client ID and Client Secret you obtained earlier. Then use curl to POST the JSON to Dataverse:

```
curl -X POST -H 'Content-type: application/json' --upload-file github.json
http://localhost:8080/api/admin/authenticationProviders
```

After restarting Glassfish you should see the new provider under “Other options” on the Log In page, as described in the [Account Creation + Management](#) section of the User Guide.

By default, the Log In page will show the “builtin” provider, but you can adjust this via the `:DefaultAuthProvider` configuration option. For details, see [Configuration](#).

## ORCID Sandbox

ORCID provides a sandbox registry, which may be useful for staging, or for development installations. This template can be used for configuring this setting (**this is not something you should use in a production environment**):

- orcid-sandbox.json

Please note that the [Prerequisites](#) section contains an step regarding CA certs in Glassfish that must be followed to get ORCID login to work.

### 4.10.3 Converting Local Users to OAuth

Once you have enabled at least one OAuth provider, existing users might want to change their login method from local to OAuth to avoid having a Dataverse-specific password. This is documented from the end user perspective in the [Account Creation + Management](#) section of the User Guide. Users will be prompted to create a new account but can choose to convert an existing local account after confirming their password.

### 4.10.4 Converting OAuth Users to Local

Whereas users convert their own accounts from local to OAuth as described above, conversion in the opposite direction is performed by a sysadmin. A common scenario may be as follows:

- A user emails Support saying, “Rather than logging in with Google, I want to log in with ORCID (or a local password). What should I do?”
- Support replies asking the user for a new email address to associate with their Dataverse account.
- The user replies with a new email address to associate with their Dataverse account.
- Support runs the curl command below, supplying the database id of the user to convert and the new email address and notes the username returned.
- Support emails the user and indicates that they should use the password reset feature to set a new password and to make sure to take note of their username under Account Information (or the password reset confirmation email) since the user never had a username before.
- The user resets password and is able to log in with their local account. All permissions have been preserved. The user can continue to log in with this Dataverse-specific password or they can convert to an identity provider, if available.

In the example below, the user has indicated that the new email address they’d like to have associated with their account is “former.oauth.user@mailinator.com” and their user id from the `authenticateduser` database table is “42”. The API token must belong to a superuser (probably the sysadmin executing the command).

```
curl -H "X-Dataverse-key: $API_TOKEN" -X PUT -d "former.oauth.user@mailinator.com" http://localhost:8080/api/admin/authenticatedUsers/id/42/convertRemoteToBuiltIn
```

The expected output is something like this:

```
{
  "status": "OK",
  "data": {
    "email": "former.oauth.user@mailinator.com",
    "username": "jdoe"
  }
}
```

Rather than looking up the user’s id in the `authenticateduser` database table, you can issue this command to get a listing of all users:

```
curl -H "X-Dataverse-key: $API_TOKEN" http://localhost:8080/api/admin/authenticatedUsers
```

Per above, you now need to tell the user to use the password reset feature to set a password for their local account.

## 4.11 External Tools

External tools can provide additional features that are not part of Dataverse itself, such as data exploration.

### Contents:

- *Inventory of External Tools*
- *Managing External Tools*
- *Building External Tools*

### 4.11.1 Inventory of External Tools

See *Inventory of External Tools*.

### 4.11.2 Managing External Tools

See the *External Tools* section of the Admin Guide.

### 4.11.3 Building External Tools

See the *Building External Tools* section of the API Guide.

## 4.12 Advanced Installation

Advanced installations are not officially supported but here we are at least documenting some tips and tricks that you might find helpful. You can find a diagram of an advanced installation in the *Preparation* section.

#### Contents:

- *Multiple Glassfish Servers*
  - *Detecting Which Glassfish Server a User Is On*

### 4.12.1 Multiple Glassfish Servers

You should be conscious of the following when running multiple Glassfish servers.

- Only one Glassfish server can be the dedicated timer server, as explained in the *Dataverse Application Timers* section of the Admin Guide.
- When users upload a logo or footer for their dataverse using the “theme” feature described in the *Dataverse Management* section of the User Guide, these logos are stored only on the Glassfish server the user happen to be on when uploading the logo. By default these logos and footers are written to the directory `/usr/local/glassfish4/glassfish/domains/domain1/docroot/logos`.
- When a sitemap is created by a Glassfish server it is written to the filesystem of just that Glassfish server. By default the sitemap is written to the directory `/usr/local/glassfish4/glassfish/domains/domain1/docroot/sitemap`.
- If Make Data Count is used, its raw logs must be copied from each Glassfish server to single instance of Counter Processor. See also the `:MDCLogPath` database setting in the *Configuration* section of this guide and the *Make Data Count* section of the Admin Guide.
- Dataset draft version logging occurs separately on each Glassfish server. See “Edit Draft Versions Logging” in the *Monitoring* section of the Admin Guide for details.
- Password aliases (`db_password_alias`, etc.) are stored per Glassfish server.

## Detecting Which Glassfish Server a User Is On

If you have successfully installed multiple Glassfish servers behind a load balancer you might like to know which server a user has landed on. A straightforward solution is to place a file called `host.txt` in a directory that is served up by Apache such as `/var/www/html` and then configure Apache not to proxy requests to `/host.txt` to Glassfish. Here are some example commands on RHEL/CentOS 7 that accomplish this:

```
[root@server1 ~]# vim /etc/httpd/conf.d/ssl.conf
[root@server1 ~]# grep host.txt /etc/httpd/conf.d/ssl.conf
ProxyPassMatch ^/host.txt !
[root@server1 ~]# systemctl restart httpd.service
[root@server1 ~]# echo $HOSTNAME > /var/www/html/host.txt
[root@server1 ~]# curl https://dataverse.example.edu/host.txt
server1.example.edu
```

You would repeat the steps above for all of your Glassfish servers. If users seem to be having a problem with a particular server, you can ask them to visit <https://dataverse.example.edu/host.txt> and let you know what they see there (e.g. “server1.example.edu”) to help you know which server to troubleshoot.

Please note that “Network Ports” under the *Configuration* section has more information on fronting Glassfish with Apache. The *Shibboleth* section talks about the use of `ProxyPassMatch`.



## DEVELOPER GUIDE

### Contents:

## 5.1 Introduction

Welcome! Dataverse is an open source project that loves contributors!

### Contents:

- *Intended Audience*
- *Getting Help*
- *Core Technologies*
- *Roadmap*
- *Kanban Board*
- *Issue Tracker*
- *Related Guides*
- *Related Projects*

### 5.1.1 Intended Audience

This guide is intended primarily for developers who want to work on the main Dataverse code base at <https://github.com/IQSS/dataverse> but see “Related Projects” below for other code you can work on!

To get started, you’ll want to set up your *Development Environment* and make sure you understand the branching strategy described in the *Version Control* section and how to make a pull request. *Testing* is expected. Opinions about *Coding Style* are welcome!

### 5.1.2 Getting Help

If you have any questions at all, please reach out to other developers via the channels listed in <https://github.com/IQSS/dataverse/blob/develop/CONTRIBUTING.md> such as <http://chat.dataverse.org> (#dataverse on freenode), the dataverse-dev mailing list, community calls, or [support@dataverse.org](mailto:support@dataverse.org).

### 5.1.3 Core Technologies

Dataverse is a Java EE application that is compiled into a war file and deployed to an application server (Glassfish) which is configured to work with a relational database (PostgreSQL) and a search engine (Solr).

We make use of a variety of Java EE technologies such as JPA, JAX-RS, JMS, and JSF. The front end is built using PrimeFaces and Bootstrap.

### 5.1.4 Roadmap

For the Dataverse development roadmap, please see <https://www.iq.harvard.edu/roadmap-dataverse-project>

### 5.1.5 Kanban Board

You can get a sense of what's currently in flight (in dev, in QA, etc.) by looking at <https://github.com/orgs/IQSS/projects/2>

### 5.1.6 Issue Tracker

We use GitHub Issues as our issue tracker: <https://github.com/IQSS/dataverse/issues>

### 5.1.7 Related Guides

If you are a developer who wants to make use of Dataverse APIs, please see the *API Guide*. If you have front-end UI questions, please see the *Style Guide*.

If you are a sysadmin who likes to code, you may be interested in hacking on installation scripts mentioned in the *Installation Guide*. We validate the installation scripts with *Tools* such as *Vagrant* and *Docker* (see the *Docker, Kubernetes, and OpenShift* section).

### 5.1.8 Related Projects

As a developer, you also may be interested in these projects related to Dataverse:

- External Tools - add additional features to Dataverse without modifying the core: *Building External Tools*
- Dataverse API client libraries - use Dataverse APIs from various languages: *Client Libraries*
- DVUploader - a stand-alone command-line Java application that uses the Dataverse API to support upload of files from local disk to a Dataset: <https://github.com/IQSS/dataverse-uploader>
- dataverse-sample-data - populate your Dataverse installation with sample data: <https://github.com/IQSS/dataverse-sample-data>
- dataverse-metrics - aggregate and visualize metrics for installations of Dataverse around the world: <https://github.com/IQSS/dataverse-metrics>
- Configuration management scripts - Ansible, Puppet, etc.: See “Advanced Installation” in the *Preparation* section of the Installation Guide.
- *Universal Numerical Fingerprint (UNF)* (Java) - a Universal Numerical Fingerprint: <https://github.com/IQSS/UNF>
- GeoConnect (Python) - create a map by uploading files to Dataverse: <https://github.com/IQSS/geoconnect>

- DataTags (Java and Scala) - tag datasets with privacy levels: <https://github.com/IQSS/DataTags>
- TwoRavens (Javascript) - a d3.js interface for exploring data and running Zelig models: <https://github.com/IQSS/TwoRavens>
- Zelig (R) - run statistical models on files uploaded to Dataverse: <https://github.com/IQSS/Zelig>
- Matrix - a visualization showing the connectedness and collaboration between authors and their affiliations.
- Third party apps - make use of Dataverse APIs: *Apps*
- chat.dataverse.org - chat interface for Dataverse users and developers: <https://github.com/IQSS/chat.dataverse.org>
- [Your project here] :)

Next: *Development Environment*

## 5.2 Development Environment

These instructions are purposefully opinionated and terse to help you get your development environment up and running as quickly as possible! Please note that familiarity with running commands from the terminal is assumed.

### Contents:

- *Quick Start*
- *Set Up Dependencies*
  - *Supported Operating Systems*
  - *Install Java*
  - *Install Netbeans or Maven*
  - *Install Homebrew (Mac Only)*
  - *Clone the Dataverse Git Repo*
  - *Build the Dataverse War File*
  - *Install jq*
  - *Install Glassfish*
    - \* *Test Glassfish Startup Time on Mac*
  - *Install PostgreSQL*
  - *Install Solr*
- *Run the Dataverse Installer Script*
  - *Verify Dataverse is Running*
- *Configure Your Development Environment for Publishing*
- *Next Steps*

## 5.2.1 Quick Start

The quickest way to get Dataverse running is to use Vagrant as described in the *Tools* section, but for day to day development work, we recommended the following setup.

## 5.2.2 Set Up Dependencies

### Supported Operating Systems

Mac OS X or Linux is required because the setup scripts assume the presence of standard Unix utilities.

Windows is not well supported, unfortunately, but Vagrant and Minishift environments are described in the *Windows Development* section.

### Install Java

Dataverse requires Java 8.

On Mac, we recommend Oracle's version of the JDK, which can be downloaded from <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

On Linux, you are welcome to use the OpenJDK available from package managers.

### Install Netbeans or Maven

NetBeans IDE (Java EE bundle) is recommended, and can be downloaded from <http://netbeans.org>. Developers may use any editor or IDE. We recommend NetBeans because it is free, works cross platform, has good support for Java EE projects, and includes a required build tool, Maven.

Below we describe how to build the Dataverse war file with Netbeans but if you prefer to use only Maven, you can find installation instructions in the *Tools* section.

### Install Homebrew (Mac Only)

On Mac, install Homebrew to simplify the steps below: <https://brew.sh>

### Clone the Dataverse Git Repo

Fork <https://github.com/IQSS/dataverse> and then clone your fork like this:

```
git clone git@github.com:[YOUR GITHUB USERNAME]/dataverse.git
```

### Build the Dataverse War File

To build the Dataverse war file using versions of Netbeans newer than 8.2 requires some setup because Java EE support is not enabled by default. An alternative is to build the war file with Maven, which is explained below.

Launch Netbeans and click “File” and then “Open Project”. Navigate to where you put the Dataverse code and double-click “dataverse” to open the project.

If you are using Netbeans 8.2, Java EE support should “just work” but if you are using a newer version of Netbeans, you will see “dataverse (broken)”. If you see “broken”, click “Tools”, “Plugins”, and “Installed”. Check the box next

to “Java Web and EE” and click “Activate”. Let Netbeans install all the dependencies. You will observe that the green “Active” checkmark does not appear next to “Java Web and EE”. Restart Netbeans.

In Netbeans, select “dataverse” under Projects and click “Run” in the menu and then “Build Project (dataverse)”. The first time you build the war file, it will take a few minutes while dependencies are downloaded from Maven Central. Feel free to move on to other steps but check back for “BUILD SUCCESS” at the end.

If you installed Maven instead of Netbeans, run `mvn package`.

NOTE: Do you use a locale different than `en_US.UTF-8` on your development machine? Are you in a different timezone than Harvard (Eastern Time)? You might experience issues while running tests that were written with these settings in mind. The Maven `pom.xml` tries to handle this for you by setting the locale to `en_US.UTF-8` and timezone UTC, but more, not yet discovered building or testing problems might lurk in the shadows.

## Install jq

On Mac, run this command:

```
brew install jq
```

On Linux, install `jq` from your package manager or download a binary from <http://stedolan.github.io/jq/>

## Install Glassfish

Glassfish 4.1 is required.

To install Glassfish, run the following commands:

```
cd /usr/local
sudo curl -O http://download.oracle.com/glassfish/4.1/release/glassfish-4.1.zip
sudo unzip glassfish-4.1.zip
sudo chown -R $USER /usr/local/glassfish4
```

## Test Glassfish Startup Time on Mac

```
cd /usr/local/glassfish4/glassfish/bin
./asadmin start-domain
grep "startup time" /usr/local/glassfish4/glassfish/domains/domain1/logs/server.log
```

If you are seeing startup times in the 30 second range (31,584ms for “Felix” for example) please be aware that startup time can be greatly reduced (to less than 1.5 seconds in our testing) if you make a small edit to your `/etc/hosts` file as described at <https://stackoverflow.com/questions/39636792/jvm-takes-a-long-time-to-resolve-ip-address-for-localhost/39698914#39698914> and <https://thoeni.io/post/macos-sierra-java/>

Look for a line that says `127.0.0.1 localhost` and add a space followed by the output of `hostname` which should be something like `foobar.local` depending on the name of your Mac. For example, the line would say `127.0.0.1 localhost foobar.local` if your Mac’s name is “foobar”.

## Install PostgreSQL

PostgreSQL 9.6 is recommended to match the version in the Installation Guide.

On Mac, go to <https://www.postgresql.org/download/macosx/> and choose “Interactive installer by EnterpriseDB” option. We’ve tested version 9.6.9. When prompted to set a password for the “database superuser (postgres)” just enter “password”.

After installation is complete, make a backup of the `pg_hba.conf` file like this:

```
sudo cp /Library/PostgreSQL/9.6/data/pg_hba.conf /Library/PostgreSQL/9.6/data/pg_hba.conf.orig
```

Then edit `pg_hba.conf` with an editor such as `vi`:

```
sudo vi /Library/PostgreSQL/9.6/data/pg_hba.conf
```

In the “METHOD” column, change all instances of “md5” to “trust”.

In the Finder, click “Applications” then “PostgreSQL 9.6” and launch the “Reload Configuration” app. Click “OK” after you see “server signaled”.

Next, launch the “pgAdmin” application from the same folder. Under “Browser”, expand “Servers” and double click “PostgreSQL 9.6”. When you are prompted for a password, leave it blank and click “OK”. If you have successfully edited “`pg_hba.conf`”, you can get in without a password.

On Linux, you should just install PostgreSQL from your package manager without worrying about the version as long as it’s 9.x. Find `pg_hba.conf` and set the authentication method to “trust” and restart PostgreSQL.

## Install Solr

Solr 7.3.1 is required.

To install Solr, execute the following commands:

```
sudo mkdir /usr/local/solr
sudo chown $USER /usr/local/solr
cd /usr/local/solr
curl -O http://archive.apache.org/dist/lucene/solr/7.3.1/solr-7.3.1.tgz
tar xvfz solr-7.3.1.tgz
cd solr-7.3.1/server/solr
cp -r configsets/_default collection1
curl -O https://raw.githubusercontent.com/IQSS/dataverse/develop/conf/solr/7.3.1/schema.xml
curl -O https://raw.githubusercontent.com/IQSS/dataverse/develop/conf/solr/7.3.1/schema_dv_mdb_fields.xml
curl -O https://raw.githubusercontent.com/IQSS/dataverse/develop/conf/solr/7.3.1/schema_dv_mdb_copies.xml
mv schema*.xml collection1/conf
curl -O https://raw.githubusercontent.com/IQSS/dataverse/develop/conf/solr/7.3.1/solrconfig.xml
mv solrconfig.xml collection1/conf/solrconfig.xml
```

```
cd /usr/local/solr/solr-7.3.1
bin/solr start
bin/solr create_core -c collection1 -d server/solr/collection1/conf
```

### 5.2.3 Run the Dataverse Installer Script

Navigate to the directory where you cloned the Dataverse git repo and run these commands:

```
cd scripts/installer
./install
```

It's fine to accept the default values.

After a while you will see `Enter admin user name [Enter to accept default]>` and you can just hit `Enter`.

#### Verify Dataverse is Running

After the script has finished, you should be able to log into Dataverse with the following credentials:

- `http://localhost:8080`
- `username: dataverseAdmin`
- `password: admin`

### 5.2.4 Configure Your Development Environment for Publishing

Run the following command:

```
curl http://localhost:8080/api/admin/settings/:DoiProvider -X PUT -d FAKE
```

This will disable DOI registration by using a fake (in-code) DOI provider. Please note that this feature is only available in version  $\geq 4.10$  and that at present, the UI will give no indication that the DOIs thus minted are fake.

### 5.2.5 Next Steps

If you can log in to Dataverse, great! If not, please see the *Troubleshooting* section. For further assistance, please see “Getting Help” in the *Introduction* section.

You're almost ready to start hacking on code. Now that the installer script has you up and running, you need to continue on to the *Tips* section to get set up to deploy code from your IDE or the command line.

---

Previous: *Introduction* | Next: *Tips*

## 5.3 Windows Development

Development on Windows is not well supported, unfortunately. You will have a much easier time if you develop on Mac or Linux as described under *Development Environment* section.

If you want to try using Windows for Dataverse development, your best best is to use Vagrant, as described below. Minishift is also an option. These instructions were tested on Windows 10.

**Contents:**

- *Running Dataverse in Vagrant*
  - *Install Vagrant*
  - *Install VirtualBox*
  - *Reboot*
  - *Install Git*
  - *Configure Git to use Unix Line Endings*
  - *Clone Git Repo*
  - *vagrant up*
- *Running Dataverse in Minishift*
  - *Install VirtualBox*
  - *Install Git*
  - *Install Minishift*
  - *Clone Git Repo*
  - *Start Minishift VM and Run Dataverse*
- *Improving Windows Support*
  - *Windows Subsystem for Linux*
  - *Discussion and Feedback*

### 5.3.1 Running Dataverse in Vagrant

#### Install Vagrant

Download and install Vagrant from <https://www.vagrantup.com>

Vagrant advises you to reboot but let's install VirtualBox first.

#### Install VirtualBox

Download and install VirtualBox from <https://www.virtualbox.org>

Note that we saw an error saying “Oracle VM VirtualBox 5.2.8 Setup Wizard ended prematurely” but then we re-ran the installer and it seemed to work.

#### Reboot

Again, Vagrant asks you to reboot, so go ahead.

#### Install Git

Download and install Git from <https://git-scm.com>



## Configure Git to use Unix Line Endings

Launch Git Bash and run the following commands:

```
git config --global core.autocrlf input
```

Pro tip: Use Shift-Insert to paste into Git Bash.

See also <https://help.github.com/articles/dealing-with-line-endings/>

If you skip this step you are likely to see the following error when you run `vagrant up`.

```
/tmp/vagrant-shell: ./install: /usr/bin/perl^M: bad interpreter: No such file or directory
```

## Clone Git Repo

From Git Bash, run the following command:

```
git clone https://github.com/IQSS/dataverse.git
```

## vagrant up

From Git Bash, run the following commands:

```
cd dataverse
```

The `dataverse` directory you changed is the one you just cloned. Vagrant will operate on a file called `Vagrantfile`.

```
vagrant up
```

After a long while you hopefully will have Dataverse installed at <http://localhost:8888>

## 5.3.2 Running Dataverse in Minishift

Minishift is a dev environment for OpenShift, which is Red Hat's distribution of Kubernetes. The *Docker, Kubernetes, and OpenShift* section contains much more detail but the essential steps for using Minishift on Windows are described below.

### Install VirtualBox

Download and install VirtualBox from <https://www.virtualbox.org>

### Install Git

Download and install Git from <https://git-scm.com>

### Install Minishift

Download Minishift from <https://docs.openshift.org/latest/minishift/getting-started/installing.html> . It should be a zip file.

From Git Bash:

```
cd ~/Downloads
```

```
unzip minishift*.zip
mkdir ~/bin
cp minishift*/minishift.exe ~/bin
```

### Clone Git Repo

From Git Bash, run the following commands:

```
git config --global core.autocrlf input
git clone https://github.com/IQSS/dataverse.git
```

### Start Minishift VM and Run Dataverse

```
minishift start --vm-driver=virtualbox --memory=8GB
eval $(minishift oc-env)
oc new-project project1
cd ~/dataverse
oc new-app conf/openshift/openshift.json
minishift console
```

This should open a web browser. In Microsoft Edge we saw `INET_E_RESOURCE_NOT_FOUND` so if you see that, try Chrome instead. A cert error is expected. Log in with the username “developer” and any password such as “asdf”.

Under “Overview” you should see a URL that has “dataverse-project1” in it. You should be able to click it and log into Dataverse with the username “dataverseAdmin” and the password “admin”.

## 5.3.3 Improving Windows Support

### Windows Subsystem for Linux

We have been unable to get Windows Subsystem for Linux (WSL) to work. We tried following the steps at <https://docs.microsoft.com/en-us/windows/wsl/install-win10> but the “Get” button was greyed out when we went to download Ubuntu.

### Discussion and Feedback

For more discussion of Windows support for Dataverse development see our community list thread “[Do you want to develop on Windows?](#)” We would be happy to incorporate feedback from Windows developers into this page. The *Writing Documentation* section describes how.

## 5.4 Tips

If you just followed the steps in *Development Environment* for the first time, you will need to get set up to deploy code to Glassfish. Below you’ll find other tips as well.

**Contents:**

- *Iterating on Code and Redeploying*
  - *Undeploy the war File from the `install` Script*
  - *Add Glassfish 4.1 as a Server in Netbeans*
  - *Ensure that Dataverse Will Be Deployed to Glassfish 4.1*
  - *Make a Small Change to the Code*
  - *Confirm the Change Was Deployed*
- *Netbeans Connector Chrome Extension*
- *Database Schema Exploration*
  - *pgAdmin*
  - *SchemaSpy*
- *Deploying With `asadmin`*
- *Running the Dataverse `install` Script in Non-Interactive Mode*
- *Preventing Glassfish from Phoning Home*
- *Solr*
- *Git*
  - *Set Up SSH Keys*
  - *Git on Mac*
  - *Automation of Custom Build Number on Webpage*
- *Sample Data*

## 5.4.1 Iterating on Code and Redeploying

When you followed the steps in the *Development Environment* section, the war file was deployed to Glassfish by the `install` script. That's fine but once you're ready to make a change to the code you will need to get comfortable with undeploying and redeploying code (a war file) to Glassfish.

It's certainly possible to manage deployment and undeployment of the war file via the command line using the `asadmin` command that ships with Glassfish (that's what the `install` script uses and the steps are documented below), but we recommend getting set up with an IDE such as Netbeans to manage deployment for you.

### Undeploy the war File from the `install` Script

Because the initial deployment of the war file was done outside of Netbeans by the `install` script, it's a good idea to undeploy that war file to give Netbeans a clean slate to work with.

Assuming you installed Glassfish in `/usr/local/glassfish4`, run the following `asadmin` command to see the version of Dataverse that the `install` script deployed:

```
/usr/local/glassfish4/bin/asadmin list-applications
```

You will probably see something like `dataverse-4.8.5 <ejb, web>` as the output. To undeploy, use whichever version you see like this:

```
/usr/local/glassfish4/bin/asadmin undeploy dataverse-4.8.5
```

Now that Glassfish doesn't have anything deployed, we can proceed with getting Netbeans set up to deploy the code.

### Add Glassfish 4.1 as a Server in Netbeans

Dataverse only works with a specific version of Glassfish (see <https://github.com/IQSS/dataverse/issues/2628>) so you need to make sure Netbeans is deploying to that version rather than a newer version of Glassfish that may have come bundled with Netbeans.

Launch Netbeans and click "Tools" and then "Servers". Click "Add Server" and select "Glassfish Server" and set the installation location to `/usr/local/glassfish4`. The default are fine so you can click "Next" and "Finish". If you are running Netbeans 8.2 and there is already a bundled version of Glassfish, you should probably remove it because (again) you need a specific version of Glassfish (4.1 as of this writing).

Please note that if you are on a Mac, Netbeans may be unable to start Glassfish due to proxy settings in Netbeans. Go to the "General" tab in Netbeans preferences and click "Test connection" to see if you are affected. If you get a green checkmark, you're all set. If you get a red exclamation mark, change "Proxy Settings" to "No Proxy" and retest. A more complicated answer having to do with changing network settings is available at <https://discussions.apple.com/thread/7680039?answerId=30715103022#30715103022> and the bug is also described at [https://netbeans.org/bugzilla/show\\_bug.cgi?id=268076](https://netbeans.org/bugzilla/show_bug.cgi?id=268076)

At this point you can manage Glassfish using Netbeans. Click "Window" and then "Services". Expand "Servers" and right-click Glassfish to stop and then start it so that it appears in the Output window. Note that you can expand "Glassfish" and "Applications" to see if any applications are deployed.

### Ensure that Dataverse Will Be Deployed to Glassfish 4.1

Click "Window" and then "Projects". Click "File" and then "Project Properties (dataverse)". Click "Run" and change "Server" from "No Server Selected" to your installation of Glassfish 4.1. Click OK.

### Make a Small Change to the Code

Let's make a tiny change to the code, compile the war file, deploy it, and verify that that we can see the change.

One of the smallest changes we can make is adjusting the build number that appears in the lower right of every page.

From the root of the git repo, run the following command to set the build number to the word "hello" (or whatever you want):

```
scripts/installer/custom-build-number hello
```

This should update or place a file at `src/main/java/BuildNumber.properties`.

Then, from Netbeans, click "Run" and then "Clean and Build Project (dataverse)". After this completes successfully, click "Run" and then "Run Project (dataverse)"

### Confirm the Change Was Deployed

After deployment, check the build number in the lower right to make sure it has been customized. You can also check the build number by running the following command:

```
curl http://localhost:8080/api/info/version
```

If you can see the change, great! Please go fix a bug or work on a feature! :)

Actually, before you start changing any code, you should create a branch as explained in the *Version Control* section.

While it's fresh in your mind, if you have any suggestions on how to make the setup of a development environment easier, please get in touch!

## 5.4.2 Netbeans Connector Chrome Extension

For faster iteration while working on JSF pages, it is highly recommended that you install the Netbeans Connector Chrome Extension listed in the *Tools* section. When you save XHTML or CSS files, you will see the changes immediately. Hipsters call this “hot reloading”. :)

## 5.4.3 Database Schema Exploration

With over 100 tables, the Dataverse PostgreSQL database (“dvndb”) can be somewhat daunting for newcomers. Here are some tips for coming up to speed. (See also the *SQL Upgrade Scripts* section.)

### pgAdmin

Back in the *Development Environment* section, we had you install pgAdmin, which can help you explore the tables and execute SQL commands. It's also listed in the *Tools* section.

### SchemaSpy

SchemaSpy is a tool that creates a website of entity-relationship diagrams based on your database.

As part of our build process for running integration tests against the latest code in the “develop” branch, we drop the database on the “phoenix” server, recreate the database by deploying the latest war file, and run SchemaSpy to create the following site: <http://phoenix.dataverse.org/schemaspy/latest/relationships.html>

To run this command on your laptop, download SchemaSpy and take a look at the syntax in `scripts/deploy/phoenix.dataverse.org/post`

To read more about the phoenix server, see the *Testing* section.

## 5.4.4 Deploying With asadmin

Sometimes you want to deploy code without using Netbeans or from the command line on a server you have ssh'ed into.

For the `asadmin` commands below, we assume you have already changed directories to `/usr/local/glassfish4/glassfish/bin` or wherever you have installed Glassfish.

There are four steps to this process:

1. Build the war file: `mvn package`
2. Check which version of Dataverse is deployed: `./asadmin list-applications`
3. Undeploy the Dataverse application (if necessary): `./asadmin undeploy dataverse-VERSION`
4. Copy the war file to the server (if necessary)
5. Deploy the new code: `./asadmin deploy /path/to/dataverse-VERSION.war`

### 5.4.5 Running the Dataverse `install` Script in Non-Interactive Mode

Rather than running the installer in “interactive” mode, it’s possible to put the values in a file. See “non-interactive mode” in the *Installation* section of the Installation Guide.

### 5.4.6 Preventing Glassfish from Phoning Home

By default, Glassfish reports analytics information. The administration guide suggests this can be disabled with `./asadmin create-jvm-options -Dcom.sun.enterprise.tools.admingui.NO_NETWORK=true`, should this be found to be undesirable for development purposes.

### 5.4.7 Solr

Once some dataverses, datasets, and files have been created and indexed, you can experiment with searches directly from Solr at <http://localhost:8983/solr/#/collection1/query> and look at the JSON output of searches, such as this wildcard search: [http://localhost:8983/solr/collection1/select?q=%3A\\*&wt=json&indent=true](http://localhost:8983/solr/collection1/select?q=%3A*&wt=json&indent=true) . You can also get JSON output of static fields Solr knows about: <http://localhost:8983/solr/collection1/schema/fields>

You can simply double-click “start.jar” rather than running `java -jar start.jar` from the command line. Figuring out how to stop Solr after double-clicking it is an exercise for the reader.

### 5.4.8 Git

#### Set Up SSH Keys

You can use git with passwords over HTTPS, but it’s much nicer to set up SSH keys. <https://github.com/settings/ssh> is the place to manage the ssh keys GitHub knows about for you. That page also links to a nice howto: <https://help.github.com/articles/generating-ssh-keys>

From the terminal, `ssh-keygen` will create new ssh keys for you:

- private key: `~/.ssh/id_rsa` - It is very important to protect your private key. If someone else acquires it, they can access private repositories on GitHub and make commits as you! Ideally, you’ll store your ssh keys on an encrypted volume and protect your private key with a password when prompted for one by `ssh-keygen`. See also “Why do passphrases matter” at <https://help.github.com/articles/generating-ssh-keys>
- public key: `~/.ssh/id_rsa.pub` - After you’ve created your ssh keys, add the public key to your GitHub account.

#### Git on Mac

On a Mac, you won’t have git installed unless you have “Command Line Developer Tools” installed but running `git clone` for the first time will prompt you to install them.

#### Automation of Custom Build Number on Webpage

You can create symbolic links from `.git/hooks/post-checkout` and `.git/hooks/post-commit` to `scripts/installer/custom-build-number-hook` to let Git automatically update `src/main/java/BuildNumber.properties` for you. This will result in showing branch name and commit id in your test deployment webpages on the bottom right corner next to the version.

When you prefer manual updates, there is another script, see above: *Make a Small Change to the Code*.

## 5.4.9 Sample Data

You may want to populate your **non-production** installation(s) of Dataverse with sample data. You have a couple options:

- Code in <https://github.com/IQSS/dataverse-sample-data> (recommended). This set of sample data includes several common data types, data subsetted from production datasets in `dataverse.harvard.edu`, datasets with file hierarchy, and more.
- Scripts called from `scripts/deploy/phoenix.dataverse.org/post`.

Previous: *Development Environment* | Next: *Troubleshooting*

## 5.5 Troubleshooting

Over in the *Development Environment* section we described the “happy path” of when everything goes right as you set up your Dataverse development environment. Here are some common problems and solutions for when things go wrong.

### Contents:

- *context-root in glassfish-web.xml Munged by Netbeans*
- *Configuring / Troubleshooting Mail Host*
- *Rebuilding Your Dev Environment*
- *DataCite*

### 5.5.1 context-root in glassfish-web.xml Munged by Netbeans

For unknown reasons, Netbeans will sometimes change the following line under `src/main/webapp/WEB-INF/glassfish-web.xml`:

```
<context-root></context-root>
```

Sometimes Netbeans will change `/` to `/dataverse`. Sometimes it will delete the line entirely. Either way, you will see very strange behavior when attempting to click around Dataverse in a browser. The homepage will load but icons will be missing. Any other page will fail to load entirely and you’ll see a Glassfish error.

The solution is to put the file back to how it was before Netbeans touched it. If anyone knows of an open Netbeans bug about this, please let us know.

### 5.5.2 Configuring / Troubleshooting Mail Host

If you have trouble with the SMTP server, consider editing the `install` script to disable the SMTP check.

Out of the box, no emails will be sent from your development environment. This is because you have to set the `:SystemEmail` setting and make sure you’ve configured your SMTP server correctly.

You can configure `:SystemEmail` like this:

```
curl -X PUT -d 'Davisverse SWAT Team <davisthedog@harvard.edu>' http://localhost:8080/api/admin/settings/:SystemEmail
```

Unfortunately for developers not at Harvard, the installer script gives you by default an SMTP server of `mail.hmdc.harvard.edu` but you can specify an alternative SMTP server when you run the installer.

You can check the current SMTP server with the `asadmin` command:

```
./asadmin get server.resources.mail-resource.mail/notifyMailSession.host
```

This command helps verify what host your domain is using to send mail. Even if it's the correct hostname, you may still need to adjust settings. If all else fails, there are some free SMTP service options available such as Gmail and MailGun. This can be configured from the GlassFish console or the command line.

1. First, navigate to your Glassfish admin console: <http://localhost:4848>
2. From the left-side panel, select **JavaMail Sessions**
3. You should see one session named **mail/notifyMailSession** – click on that.

From this window you can modify certain fields of your Dataverse's `notifyMailSession`, which is the JavaMail session for outgoing system email (such as on user signup or data publication). Two of the most important fields we need are:

- **Mail Host:** The DNS name of the default mail server (e.g. `smtp.gmail.com`)
- **Default User:** The username provided to your Mail Host when you connect to it (e.g. `john.doe@gmail.com`)

Most of the other defaults can safely be left as is. **Default Sender Address** indicates the address that your installation's emails are sent from.

If your user credentials for the SMTP server require a password, you'll need to configure some **Additional Properties** at the bottom.

**IMPORTANT:** Before continuing, it's highly recommended that your Default User account does NOT use a password you share with other accounts, as one of the additional properties includes entering the Default User's password (without concealing it on screen). For `smtp.gmail.com` you can safely use an [app password](#) or create an extra Gmail account for use with your Dataverse dev environment.

Authenticating yourself to a Mail Host can be tricky. As an example, we'll walk through setting up our JavaMail Session to use `smtp.gmail.com` as a host by way of SSL on port 465. Use the Add Property button to generate a blank property for each name/value pair.

Name	Value
<code>mail.smtp.auth</code>	<code>true</code>
<code>mail.smtp.password</code>	[user's ( <i>app</i> ) password*]
<code>mail.smtp.port</code>	<code>465</code>
<code>mail.smtp.socketFactory.port</code>	<code>465</code>
<code>mail.smtp.socketFactory.fallback</code>	<code>false</code>
<code>mail.smtp.socketFactory.class</code>	<code>javax.net.ssl.SSLSocketFactory</code>

**\*WARNING:** Entering a password here will *not* conceal it on-screen. It's recommended to use an *app password* (for `smtp.gmail.com` users) or utilize a dedicated/non-personal user account with SMTP server auths so that you do not risk compromising your password.

Save these changes at the top of the page and restart your Glassfish server to try it out.

The mail session can also be set from command line. To use this method, you will need to delete your `notifyMailSession` and create a new one. See the below example:

- Delete: `./asadmin delete-javamail-resource mail/MyMailSession`
- Create (remove brackets and replace the variables inside): `./asadmin create-javamail-resource --mailhost [smtp.gmail.com] --mailuser [test@test.com] --fromaddress [test@test.com] --property mail.smtp.auth=[true]:mail.smtp.password=[password]:mail.smtp.port=[465]:mail.smtp.socketFactory.port=[465]:mail.smtp.socketFactory.fallback=[false]:mail.smtp.`



```
socketFactory.class=[javax.net.ssl.SSLSocketFactory] mail/
notifyMailSession
```

These properties can be tailored to your own preferred mail service, but if all else fails these settings work fine with Dataverse development environments for your localhost.

- If you're seeing a "Relay access denied" error in your Glassfish logs when your app attempts to send an email, double check your user/password credentials for the Mail Host you're using.
- If you're seeing a "Connection refused" / similar error upon email sending, try another port.

As another example, here is how to create a Mail Host via command line for Amazon SES:

- Delete: `./asadmin delete-javamail-resource mail/MyMailSession`
- Create (remove brackets and replace the variables inside): `./asadmin create-javamail-resource --mailhost email-smtp.us-east-1.amazonaws.com --mailuser [test\@test\.com] --fromaddress [test\@test\.com] --transprotocol aws --transprotocolclass com.amazonaws.services.simpleemail.AWSJavaMailTransport --property mail.smtp.auth=true:mail.smtp.user=[aws_access_key]:mail.smtp.password=[aws_secret_key]:mail.transport.protocol=smtp:mail.smtp.port=587:mail.smtp.starttls.enable=true mail/notifyMailSession`

### 5.5.3 Rebuilding Your Dev Environment

If you have an old copy of the database and old Solr data and want to start fresh, here are the recommended steps:

- drop your old database
- clear out your existing Solr index: `scripts/search/clear`
- run the installer script above - it will create the db, deploy the app, populate the db with reference data and run all the scripts that create the domain metadata fields. You no longer need to perform these steps separately.
- confirm you are using the latest Dataverse-specific Solr schema.xml and included XML files (schema\_dv\_cmb\_[copies]fields.xml)
- confirm `http://localhost:8080` is up
- If you want to set some dataset-specific facets, go to the root dataverse (or any dataverse; the selections can be inherited) and click "General Information" and make choices under "Select Facets". There is a ticket to automate this: <https://github.com/IQSS/dataverse/issues/619>

You may also find <https://github.com/IQSS/dataverse/blob/develop/scripts/deploy/phoenix.dataverse.org/deploy> and related scripts interesting because they demonstrate how we have at least partially automated the process of tearing down a Dataverse installation and having it rise again, hence the name "phoenix." See also "Fresh Reinstall" in the *Installation* section of the Installation Guide.

### 5.5.4 DataCite

If you are seeing Response code: 400, [url] domain of URL is not allowed it's probably because your `dataverse.siteUrl JVM` option is unset or set to localhost (`-Ddataverse.siteUrl=http://localhost:8080`). You can try something like this:

```
./asadmin delete-jvm-options '-Ddataverse.siteUrl=http://localhost:8080'
./asadmin create-jvm-options '-Ddataverse.siteUrl=http://demo.dataverse.org'
```

Previous: *Tips* | Next: *Version Control*

## 5.6 Version Control

The Dataverse Project uses git for version control and GitHub for hosting. On this page we'll explain where to find the code, our branching strategy, advice on how to make a pull request, and other git tips.

### Contents:

- *Where to Find the Dataverse Code*
- *Branching Strategy*
  - *Goals*
  - *Branches*
    - \* *The “master” Branch*
    - \* *The “develop” Branch*
    - \* *Feature Branches*
- *How to Make a Pull Request*
  - *Find or Create a GitHub Issue*
  - *Create a New Branch off the develop Branch*
  - *Commit Your Change to Your New Branch*
  - *Push Your Branch to GitHub*
  - *Make a Pull Request*
  - *Make Sure Your Pull Request Has Been Advanced to Code Review*
- *How to Resolve Conflicts in Your Pull Request*
- *Adding Commits to a Pull Request from a Fork*

### 5.6.1 Where to Find the Dataverse Code

The main Dataverse code at <https://github.com/IQSS/dataverse> but as explained in the *Introduction* section under “Related Projects”, there are many other code bases you can hack on if you wish!

### 5.6.2 Branching Strategy

#### Goals

The goals of the Dataverse branching strategy are:

- allow for concurrent development
- only ship stable code

We follow a simplified “git flow” model described at <http://nvie.com/posts/a-successful-git-branching-model/> involving a “master” branch, a “develop” branch, and feature branches such as “1234-bug-fix”.

## Branches

### The “master” Branch

The “master” branch represents released versions of Dataverse. As mentioned in the *Making Releases* section, at release time we update the master branch to include all the code for that release. Commits are never made directly to master. Rather, master is updated only when we merge code into it from the “develop” branch.

### The “develop” Branch

The “develop” branch represents code that was stable enough to merge from a “feature” branch (described below) and that will make it into the next release. Like master, commits are never made to the develop branch. The develop branch is where integration occurs. Your goal is have your code merged into the develop branch after it has been reviewed.

## Feature Branches

Feature branches are used for both developing features and fixing bugs. They are named after the GitHub issue they are meant to address, so create a GitHub issue if you need to.

“3728-doc-apipolicy-fix” is an example of a fine name for your feature branch. It tells us that you are addressing <https://github.com/IQSS/dataverse/issues/3728> and the “slug” is short, descriptive, and starts with the issue number.

## 5.6.3 How to Make a Pull Request

Pull requests take all shapes and sizes, from a one-character typo fix to hundreds of files changing at once. Generally speaking, smaller pull requests are better so that they are easier to code review. That said, don’t hold back on writing enough code or documentation to address the issue to the best of your ability.

If you are writing code (rather than documentation), please see *Testing* for guidance on writing tests.

The example of creating a pull request below has to do with fixing an important issue with the documentation but applies to fixing code as well.

### Find or Create a GitHub Issue

For guidance on which issue to work on, please ask! Also, see <https://github.com/IQSS/dataverse/blob/develop/CONTRIBUTING.md>

Let’s say you want to tackle <https://github.com/IQSS/dataverse/issues/3728> which points out a typo in a page of Dataverse’s documentation.

If you tell us your GitHub username we are happy to add you to the “read only” team at <https://github.com/orgs/IQSS/teams/dataverse-readonly/members> so that we can assign the issue to you while you’re working on it. You can also tell us if you’d like to be added to the *Dataverse Community Contributors spreadsheet*.

### Create a New Branch off the develop Branch

Always create your feature branch from the latest code in develop, pulling the latest code if necessary. As mentioned above, your branch should have a name like “3728-doc-apipolicy-fix” that starts with the issue number you are addressing, and ends with a short, descriptive name. Dashes (“-”) and underscores (“\_”) in your branch name are ok, but please try to avoid other special characters such as ampersands (“&”) than have special meaning in Unix shells.

### Commit Your Change to Your New Branch

Making a commit (or several commits) to that branch. Ideally the first line of your commit message includes the number of the issue you are addressing, such as `Fixed BlockedApiPolicy #3728`.

### Push Your Branch to GitHub

Push your feature branch to your fork of Dataverse. Your git command may look something like `git push origin 3728-doc-apipolicy-fix`.

### Make a Pull Request

Make a pull request to get approval to merge your changes into the develop branch. Note that once a pull request is created, we'll remove the corresponding issue from our kanban board so that we're only tracking one card.

Feedback on the pull request template we use is welcome! Here's an example of a pull request for issue #3827: <https://github.com/IQSS/dataverse/pull/3827>

### Make Sure Your Pull Request Has Been Advanced to Code Review

Now that you've made your pull request, your goal is to make sure it appears in the "Code Review" column at <https://github.com/orgs/IQSS/projects/2>.

Look at <https://github.com/IQSS/dataverse/blob/master/CONTRIBUTING.md> for various ways to reach out to developers who have enough access to the GitHub repo to move your issue and pull request to the "Code Review" column.

## 5.6.4 How to Resolve Conflicts in Your Pull Request

Unfortunately, pull requests can quickly become "stale" and unmergeable as other pull requests are merged into the develop branch ahead of you. This is completely normal, and often occurs because other developers made their pull requests before you did.

The Dataverse team may ping you to ask you to merge the latest from the develop branch into your branch and resolve merge conflicts. If this sounds daunting, please just say so and we will assist you.

If you'd like to resolve the merge conflicts yourself, here are some steps to do so that make use of GitHub Desktop and Netbeans.

#### In GitHub Desktop:

1. Sync from develop.
2. Open the specific branch that's having the merge conflict.
3. Click "Update from develop".

#### In Netbeans:

4. Click Window -> Favorites and open your local Dataverse project folder in the Favorites panel.
5. In this file browser, you can follow the red cylinder icon to find files with merge conflicts.
6. Double click the red merge conflicted file.
7. Right click on the red tab for that file and select Git -> Resolve Conflicts.
8. Resolve on right or left (if you select "both" you can do finer edits after).

9. Save all changes

**In GitHub Desktop:**

10. Commit the merge (append issue number to end, e.g. #3728) and leave note about what was resolved.

**In GitHub Issues:**

11. Leave a comment for the Dataverse team that you have resolved the merge conflicts.

## 5.6.5 Adding Commits to a Pull Request from a Fork

By default, when a pull request is made from a fork, “Allow edits from maintainers” is checked as explained at <https://help.github.com/articles/allowing-changes-to-a-pull-request-branch-created-from-a-fork/>

This is a nice feature of GitHub because it means that the core dev team for Dataverse can make small (or even large) changes to a pull request from a contributor to help the pull request along on its way to QA and being merged.

GitHub documents how to make changes to a fork at <https://help.github.com/articles/committing-changes-to-a-pull-request-branch-created-from-a-fork/> but as of this writing the steps involve making a new clone of the repo. This works but you might find it more convenient to add a “remote” to your existing clone. The example below uses the fork at <https://github.com/OdumInstitute/dataverse> and the branch `4709-postgresql_96` but the technique can be applied to any fork and branch:

```
git remote add OdumInstitute git@github.com:OdumInstitute/dataverse.git
git fetch OdumInstitute
git checkout 4709-postgresql_96
vim path/to/file.txt
git commit
git push OdumInstitute 4709-postgresql_96
```

Previous: *Troubleshooting* | Next: *SQL Upgrade Scripts*

## 5.7 SQL Upgrade Scripts

The database schema for Dataverse is constantly evolving and we have adopted a tool called Flyway to help keep your development environment up to date and in working order. As you make changes to the database schema (changes to @Entity classes), you must write SQL upgrade scripts when needed and follow Flyway file naming conventions.

**Contents:**

- *Location of SQL Upgrade Scripts*
- *How to Determine if You Need to Create a SQL Upgrade Script*
- *How to Create a SQL Upgrade Script*
- *Troubleshooting*
  - *Renaming SQL Upgrade Scripts*

### 5.7.1 Location of SQL Upgrade Scripts

`src/main/resources/db/migration` is the directory where we keep SQL upgrade scripts for Flyway to find. In the past (before adopting Flyway) we used to keep SQL upgrade scripts in `scripts/database/upgrades`. These scripts can still be used as reference but no new scripts should be added there.

### 5.7.2 How to Determine if You Need to Create a SQL Upgrade Script

If you are creating a new database table (which maps to an `@Entity` in JPA), you do not need to create or update a SQL upgrade script. The reason for this is that we use `create-tables` in `src/main/resources/META-INF/persistence.xml` so that new tables are automatically created by Glassfish when you deploy your war file.

If you are doing anything other than creating a new database table such as adding a column to an existing table, you must create or update a SQL upgrade script.

### 5.7.3 How to Create a SQL Upgrade Script

We assume you have already read the *Version Control* section and have been keeping your feature branch up to date with the “develop” branch.

Create a new file called something like `V4.11.0.1__5565-sanitize-directory-labels.sql` in the `src/main/resources/db/migration` directory. Use a version like “4.11.0.1” in the example above where the previously released version was 4.11, ensuring that the version number is unique. Note that this is not the version that you expect the code changes to be included in (4.12 in this example). For the “description” you should use the name of your branch, which should include the GitHub issue you are working on, as in the example above. To read more about Flyway file naming conventions, see <https://flywaydb.org/documentation/migrations#naming>

The SQL migration script you wrote will be part of the war file and executed when the war file is deployed. To see a history of Flyway database migrations that have been applied, look at the `flyway_schema_history` table.

As with any task related to Dataverse development, if you need any help writing SQL upgrade scripts, please reach out using any of the channels mentioned under “Getting Help” in the *Introduction* section.

### 5.7.4 Troubleshooting

#### Renaming SQL Upgrade Scripts

Please note that if you need to rename your script (because a new version of Dataverse was released, for example), you will see the error “FlywayException: Validate failed: Detected applied migration not resolved locally” when you attempt to deploy and deployment will fail.

To resolve this problem, delete the old migration from the `flyway_schema_history` table and attempt to re-deploy.

---

Previous: *Version Control* | Next: *Testing*

## 5.8 Testing

In order to keep our codebase healthy, the Dataverse project encourages developers to write automated tests in the form of unit tests and integration tests. We also welcome ideas for how to improve our automated testing.

**Contents:**

- *The Health of a Codebase*
- *Testing in Depth*
- *Unit Tests*
  - *Unit Test Automation Overview*
  - *Writing Unit Tests with JUnit*
    - \* *Refactoring Code to Make It Unit-Testable*
    - \* *Parameterized Tests and JUnit Theories*
    - \* *Observing Changes to Code Coverage*
    - \* *Testing Commands*
    - \* *Running Non-Essential (Excluded) Unit Tests*
- *Integration Tests*
  - *Running the full API test suite using Docker*
  - *Getting Set Up to Run REST Assured Tests*
    - \* *The Burrito Key*
    - \* *Root Dataverse Permissions*
    - \* *Publish Root Dataverse*
    - \* *dataverse.siteUrl*
    - \* *Identifier Generation*
  - *Writing Integration Tests with REST Assured*
- *Measuring Coverage of Integration Tests*
  - *Add jacocoagent.jar to Glassfish*
  - *Add jacococli.jar to the WAR File*
  - *Deploy the Instrumented WAR File*
  - *Run Integration Tests*
  - *Create Code Coverage Report*
  - *Read Code Coverage Report*
- *Load/Performance Testing*
  - *Locust*
  - *download-files.sh script*
- *Continuous Integration*
  - *Enhance build time by caching dependencies*
  - *The Phoenix Server*
    - \* *How the Phoenix Tests Work*
    - \* *How to Run the Phoenix Tests*

*\* List of Tests Run Against the Phoenix Server*

- *Accessibility Testing*
  - *Accessibility Policy*
  - *Accessibility Tools*
- *Future Work*
  - *Future Work on Unit Tests*
  - *Future Work on Integration Tests*
  - *Browser-Based Testing*
  - *Installation Testing*
  - *Future Work on Load/Performance Testing*
  - *Future Work on Accessibility Testing*

### 5.8.1 The Health of a Codebase

Before we dive into the nut and bolts of testing, let's back up for a moment and think about why we write automated tests in the first place. Writing automated tests is an investment and leads to better quality software. Counterintuitively, writing tests and executing them regularly allows a project to move faster. Martin Fowler explains this well while talking about the health of a codebase:

“This is an economic judgment. Several times, many times, I run into teams that say something like, ‘Oh well. Management isn’t allowing us to do a quality job here because it will slow us down. And we’ve appealed to management and said we need to put more quality in the code, but they’ve said no, we need to go faster instead.’ And my comment to that is well, as soon as you’re framing it in terms of code quality versus speed, you’ve lost. Because the whole point of refactoring is to go faster.

“And this is why I quite like playing a bit more with the metaphor as the health of a codebase. If you keep yourself healthy then you’ll be able to run faster. But if you just say, ‘Well, I want to run a lot so I’m therefore going to run a whole load all the time and not eat properly and not pay attention about this shooting pain going up my leg,’ then you’re not going to be able to run quickly very long. **You have to pay attention to your health. And same with the codebase. You have to continuously say, ‘How do we keep it in a healthy state? Then we can go fast,’ because we’re running marathons here with codebases. And if we neglect that internal quality of the codebase, it hits you surprisingly fast.**“

—Martin Fowler at

<https://devchat.tv/ruby-rogues/178-rr-book-club-refactoring-ruby-with-martin-fowler>

### 5.8.2 Testing in Depth

Security in depth might mean that your castle has a moat as well as high walls. Likewise, when testing, you should consider testing a various layers of the stack using both unit tests and integration tests.

When writing tests, you may find it helpful to first map out which functions of your code you want to test, and then write a functional unit test for each which can later comprise a larger integration test.

### 5.8.3 Unit Tests

Creating unit tests for your code is a helpful way to test what you’ve built piece by piece.



Unit tests can be executed without runtime dependencies on PostgreSQL, Solr, or any other external system. They are the lowest level of testing and are executed constantly on developers' laptops as part of the build process and via continuous integration services in the cloud.

A unit test should execute an operation of your code in a controlled fashion. You must make an assertion of what the expected response gives back. It's important to test optimistic output and assertions (the "happy path"), as well as unexpected input that leads to failure conditions. Know how your program should handle anticipated errors/exceptions and confirm with your test(s) that it does so properly.

## Unit Test Automation Overview

We use a variety of tools to write, execute, and measure the code coverage of unit tests, including Maven, JUnit, Jacoco, GitHub, Travis, and Coveralls. We'll explain the role of each tool below, but here's an overview of what you can expect from the automation we've set up.

As you prepare to make a pull request, as described in the *Version Control* section, you will be working on a new branch you create from the "develop" branch. Let's say your branch is called `1012-private-url`. As you work, you are constantly invoking Maven to build the war file. When you do a "clean and build" in Netbeans, Maven runs all the unit tests (anything ending with `Test.java`) and then runs the results through a tool called Jacoco that calculates code coverage. When you push your branch to GitHub and make a pull request, a web service called Travis CI runs Maven and Jacoco on your branch and pushes the results to Coveralls, which is a web service that tracks changes to code coverage over time.

To make this more concrete, observe that <https://github.com/IQSS/dataverse/pull/3111> has comments from a GitHub user called `coveralls` saying things like "Coverage increased (+0.5%) to 5.547% when pulling `dd6ceb1` on `1012-private-url` into `be5b26e` on `develop`." Clicking on the comment should lead you to a URL such as <https://coveralls.io/builds/7013870> which shows how code coverage has gone up or down. That page links to a page such as <https://travis-ci.org/IQSS/dataverse/builds/144840165> which shows the build on the Travis side that pushed the results to Coveralls.

The main takeaway should be that we care about unit testing enough to measure the changes to code coverage over time using automation. Now let's talk about how you can help keep our code coverage up by writing unit tests with JUnit.

## Writing Unit Tests with JUnit

We are aware that there are newer testing tools such as TestNG, but we use JUnit because it's tried and true.

If writing tests is new to you, poke around existing unit tests which all end in `Test.java` and live under `src/test`. Each test is annotated with `@Test` and should have at least one assertion which specifies the expected result. In Netbeans, you can run all the tests in it by clicking "Run" -> "Test File". From the test file, you should be able to navigate to the code that's being tested by right-clicking on the file and clicking "Navigate" -> "Go to Test/Tested class". Likewise, from the code, you should be able to use the same "Navigate" menu to go to the tests.

NOTE: Please remember when writing tests checking possibly localized outputs to check against `en_US.UTF-8` and `UTC` `110n` strings!

## Refactoring Code to Make It Unit-Testable

Existing code is not necessarily written in a way that lends itself to easy testing. Generally speaking, it is difficult to write unit tests for both JSF "backing" beans (which end in `Page.java`) and "service" beans (which end in `Service.java`) because they require the database to be running in order to test them. If service beans can be exercised via API they can be tested with integration tests (described below) but a good technique for making the

logic testable it to move code to “util beans” (which end in `Util.java`) that operate on Plain Old Java Objects (POJOs). `PrivateUrlUtil.java` is a good example of moving logic from `PrivateUrlServiceBean.java` to a “util” bean to make the code testable.

### Parameterized Tests and JUnit Theories

Often times you will want to test a method multiple times with similar values. In order to avoid test bloat (writing a test for every data combination), JUnit offers Data-driven unit tests with `Parameterized.class` and `Theories.class`. This allows a test to be run for each set of defined data values. For reference, take a look at issue <https://github.com/IQSS/dataverse/issues/5619>.

### Observing Changes to Code Coverage

Once you’ve written some tests, you’re probably wondering how much you’ve helped to increase the code coverage. In Netbeans, do a “clean and build.” Then, under the “Projects” tab, right-click “dataverse” and click “Code Coverage” -> “Show Report”. For each Java file you have open, you should be able to see the percentage of code that is covered by tests and every line in the file should be either green or red. Green indicates that the line is being exercised by a unit test and red indicates that it is not.

In addition to seeing code coverage in Netbeans, you can also see code coverage reports by opening `target/site/jacoco/index.html` in your browser.

### Testing Commands

You might find studying the following test classes helpful in writing tests for commands:

- `CreatePrivateUriCommandTest.java`
- `DeletePrivateUriCommandTest.java`
- `GetPrivateUriCommandTest.java`

In addition, there is a writeup on “The Testable Command” at <https://github.com/IQSS/dataverse/blob/develop/doc/theTestableCommand/TheTestableCommand.md>.

### Running Non-Essential (Excluded) Unit Tests

You should be aware that some unit tests have been deemed “non-essential” and have been annotated with `@Category(NonEssentialTests.class)` and are excluded from the “dev” Maven profile, which is the default profile. All unit tests (that have not been annotated with `@Ignore`), including these non-essential tests, are run from continuous integration systems such as Jenkins and Travis CI with the following `mvn` command that invokes a non-default profile:

```
mvn test -P all-unit-tests
```

Typically <https://travis-ci.org/IQSS/dataverse> will show a higher number of unit tests executed because it uses the profile above.

Generally speaking, unit tests have been flagged as non-essential because they are slow or because they require an Internet connection. You should not feel obligated to run these tests continuously but you can use the `mvn` command above to run them. To iterate on the unit test in Netbeans and execute it with “Run -> Test File”, you must temporarily comment out the annotation flagging the test as non-essential.

## 5.8.4 Integration Tests

Unit tests are fantastic for low level testing of logic but aren't especially real-world-applicable because they do not exercise Dataverse as it runs in production with a database and other runtime dependencies. We test in-depth by also writing integration tests to exercise a running system.

Unfortunately, the term “integration tests” can mean different things to different people. For our purposes, an integration test has the following qualities:

- Integration tests exercise Dataverse APIs.
- Integration tests are not automatically on developers' laptops.
- Integration tests operate on an installation of Dataverse that is running and able to talk to both PostgreSQL and Solr.
- Integration tests are written using REST Assured.

### Running the full API test suite using Docker

To run the full suite of integration tests on your laptop, we recommend using the “all in one” Docker configuration described in `conf/docker-aio/readme.txt` in the root of the repo.

Alternatively, you can run tests against Glassfish running on your laptop by following the “getting set up” steps below.

### Getting Set Up to Run REST Assured Tests

Unit tests are run automatically on every build, but dev environments and servers require special setup to run REST Assured tests. In short, Dataverse needs to be placed into an insecure mode that allows arbitrary users and datasets to be created and destroyed. This differs greatly from the out-of-the-box behavior of Dataverse, which we strive to keep secure for sysadmins installing the software for their institutions in a production environment.

The *Development Environment* section currently refers developers here for advice on getting set up to run REST Assured tests, but we'd like to add some sort of “dev” flag to the installer to put Dataverse in “insecure” mode, with lots of scary warnings that this dev mode should not be used in production.

The instructions below assume a relatively static dev environment on a Mac. There is a newer “all in one” Docker-based approach documented in the *Docker, Kubernetes, and OpenShift* section under “Docker” that you may like to play with as well.

### The Burrito Key

For reasons that have been lost to the mists of time, Dataverse really wants you to have a burrito. Specifically, if you're trying to run REST Assured tests and see the error “Dataverse config issue: No API key defined for built in user management”, you must run the following curl command (or make an equivalent change to your database):

```
curl -X PUT -d 'burrito' http://localhost:8080/api/admin/settings/  
BuiltinUsers.KEY
```

Without this “burrito” key in place, REST Assured will not be able to create users. We create users to create objects we want to test, such as dataverses, datasets, and files.

### Root Dataverse Permissions

In your browser, log in as `dataverseAdmin` (password: `admin`) and click the “Edit” button for your root dataverse. Navigate to Permissions, then the Edit Access button. Under “Who can add to this dataverse?” choose “Anyone with a dataverse account can add sub dataverses” if it isn’t set to this already.

Alternatively, this same step can be done with this script: `scripts/search/tests/grant-authusers-add-on-root`

### Publish Root Dataverse

The root dataverse must be published for some of the REST Assured tests to run.

### `dataverse.siteUrl`

When run locally (as opposed to a remote server), some of the REST Assured tests require the `dataverse.siteUrl` JVM option to be set to `http://localhost:8080`. See “JVM Options” under the [Configuration](#) section of the Installation Guide for advice changing JVM options. First you should check to check your JVM options with:

```
./asadmin list-jvm-options | egrep 'dataverse|doi'
```

If `dataverse.siteUrl` is absent, you can add it with:

```
./asadmin create-jvm-options "-Ddataverse.siteUrl=http://localhost:8080"
```

### Identifier Generation

`DatasetsIT.java` exercises the feature where the “identifier” of a DOI can be a digit and requires a sequence to be added to your database. See `IdentifierGenerationStrategy` under the [Configuration](#) section for adding this sequence to your installation of PostgreSQL.

### Writing Integration Tests with REST Assured

Before writing any new REST Assured tests, you should get the tests to pass in an existing REST Assured test file. `BuiltinUsersIT.java` is relatively small and requires less setup than other test files.

You do not have to reinvent the wheel. There are many useful methods you can call in your own tests – especially within `UtilIT.java` – when you need your test to create and/or interact with generated accounts, files, datasets, etc. Similar methods can subsequently delete them to get them out of your way as desired before the test has concluded.

For example, if you’re testing your code’s operations with user accounts, the method `UtilIT.createRandomUser()`; can generate an account for your test to work with. The same account can then be deleted by your program by calling the `UtilIT.deleteUser()`; method on the imaginary friend your test generated.

Remember, it’s only a test (and it’s not graded)! Some guidelines to bear in mind:

- Map out which logical functions you want to test
- Understand what’s being tested and ensure it’s repeatable
- Assert the conditions of success / return values for each operation \* A useful resource would be [HTTP status codes](#)
- Let the code do the labor; automate everything that happens when you run your test file.

- Just as with any development, if you're stuck: ask for help!

To execute existing integration tests on your local Dataverse, a helpful command line tool to use is [Maven](#). You should have Maven installed as per the [Development Environment](#) guide, but if not it's easily done via Homebrew: `brew install maven`.

Once installed, you may run commands with `mvn [options] [<goal(s)>] [<phase(s)>]`.

- If you want to run just one particular API test, it's as easy as you think:

```
mvn test -Dtest=FileRecordJobIT
```

- To run more than one test at a time, separate by commas:

```
mvn test -Dtest=FileRecordJobIT,ConfirmEmailIT
```

- To run any test(s) on a particular domain, replace localhost:8080 with desired domain name:

```
mvn test -Dtest=FileMetadataIT -Ddataverse.test.baseurl='http://localhost:8080'
```

To see the full list of tests used by the Docker option mentioned above, see `run-test-suite.sh`.

### 5.8.5 Measuring Coverage of Integration Tests

Measuring the code coverage of integration tests with Jacoco requires several steps. In order to make these steps clear we'll use `"/usr/local/glassfish4"` as the Glassfish directory and `"glassfish"` as the Glassfish Unix user.

#### Add jacocoagent.jar to Glassfish

In order to get code coverage reports out of Glassfish we'll be adding `jacocoagent.jar` to the Glassfish `"lib"` directory.

First, we need to download Jacoco. Look in `pom.xml` to determine which version of Jacoco we are using. As of this writing we are using 0.8.1 so in the example below we download the Jacoco zip from <https://github.com/jacoco/jacoco/releases/tag/v0.8.1>

Note that we are running the following commands as the user `"glassfish"`. In short, we stop Glassfish, add the Jacoco jar file, and start up Glassfish again.

```
su - glassfish
cd /home/glassfish
mkdir -p local/jacoco-0.8.1
cd local/jacoco-0.8.1
wget https://github.com/jacoco/jacoco/releases/download/v0.8.1/jacoco-0.8.1.zip
unzip jacoco-0.8.1.zip
/usr/local/glassfish4/bin/asadmin stop-domain
cp /home/glassfish/local/jacoco-0.8.1/lib/jacocoagent.jar /usr/local/glassfish4/
↪glassfish/lib
/usr/local/glassfish4/bin/asadmin start-domain
```

#### Add jacococli.jar to the WAR File

As the `"glassfish"` user download `instrument_war_jacoco.bash` (or skip ahead to the `"git clone"` step to get the script that way) and give it two arguments:

- path to your pristine WAR file
- path to the new WAR file the script will create with `jacococli.jar` in it

```
./instrument_war_jacoco.bash dataverse.war dataverse-jacoco.war
```

### Deploy the Instrumented WAR File

Please note that you'll want to undeploy the old WAR file first, if necessary.

Run this as the “glassfish” user.

```
/usr/local/glassfish4/bin/asadmin deploy dataverse-jacoco.war
```

Note that after deployment the file “/usr/local/glassfish4/glassfish/domains/domain1/config/jacoco.exec” exists and is empty.

### Run Integration Tests

Note that even though you see “docker-ai0” in the command below, we assume you are not necessarily running the test suite within Docker. (Some day we'll probably move this script to another directory.) For this reason, we pass the URL with the normal port (8080) that Glassfish runs on to the `run-test-suite.sh` script.

Note that “/usr/local/glassfish4/glassfish/domains/domain1/config/jacoco.exec” will become non-empty after you stop and start Glassfish. You must stop and start Glassfish before every run of the integration test suite.

```
/usr/local/glassfish4/bin/asadmin stop-domain
/usr/local/glassfish4/bin/asadmin start-domain
git clone https://github.com/IQSS/dataverse.git
cd dataverse
conf/docker-ai0/run-test-suite.sh http://localhost:8080
```

(As an aside, you are not limited to API tests for the purposes of learning which code paths are being executed. You could click around the GUI, for example. Jacoco doesn't know or care how you exercise the application.)

### Create Code Coverage Report

Run these commands as the “glassfish” user. The `cd dataverse` means that you should change to the directory where you cloned the “dataverse” git repo.

```
cd dataverse
java -jar /home/glassfish/local/jacoco-0.8.1/lib/jacococli.jar report --classfiles_
↪target/classes --sourcefiles src/main/java --html target/coverage-it/ /usr/local/
↪glassfish4/glassfish/domains/domain1/config/jacoco.exec
```

### Read Code Coverage Report

`target/coverage-it/index.html` is the place to start reading the code coverage report you just created.

## 5.8.6 Load/Performance Testing

### Locust

Load and performance testing is conducted on an as-needed basis but we're open to automating it. As of this writing Locust ( <https://locust.io> ) scripts at [https://github.com/IQSS/dataverse-helper-scripts/tree/master/src/stress\\_tests](https://github.com/IQSS/dataverse-helper-scripts/tree/master/src/stress_tests) have been used.

### download-files.sh script

One way of generating load is by downloading many files. You can download `download-files.sh`, make it executable (`chmod 755`), and run it with `--help`. You can use `-b` to specify the base URL of the Dataverse installation and `-s` to specify the number of seconds to wait between requests like this:

```
./download-files.sh -b https://dev1.dataverse.org -s 2
```

The script requires a file called `files.txt` to operate and database IDs for the files you want to download should each be on their own line.

## 5.8.7 Continuous Integration

The Dataverse Project currently makes use of two Continuous Integration platforms, Travis and Jenkins.

Travis builds are configured via `.travis.yml` and a *GitHub webhook* <<https://docs.travis-ci.com/user/notifications/#configuring-webhook-notifications>>; build output is viewable at <https://travis-ci.org/IQSS/dataverse/builds>

Our Jenkins config is a work in progress and may be viewed at <https://github.com/IQSS/dataverse-jenkins/> A corresponding GitHub webhook is required. Build output is viewable at <https://jenkins.dataverse.org/>

As always, pull requests to improve our continuous integration configurations are welcome.

### Enhance build time by caching dependencies

In the future, CI builds in ephemeral build environments and Docker builds can benefit from caching all dependencies and plugins. As Dataverse is a huge project, build times can be enhanced by avoiding re-downloading everything when the Maven POM is unchanged. To seed the cache, use the following Maven goal before using Maven in (optional) offline mode in your scripts:

```
mvn de.qaware.maven:go-offline-maven-plugin:resolve-dependencies``
mvn -o package -DskipTests
```

The example above builds the WAR file without running any tests. For other scenarios: not using offline mode allows Maven to download more dynamic dependencies, which are not easy to track, like Surefire Plugins. Overall downloads will reduced anyway.

You will obviously have to utilize caching functionality of your CI service or do proper Docker layering.

## The Phoenix Server

### How the Phoenix Tests Work

A server at <http://phoenix.dataverse.org> has been set up to test the latest code from the develop branch. Testing is done using chained builds of Jenkins jobs:

- A war file is built from the latest code in develop: <https://build.hmdc.harvard.edu:8443/job/phoenix.dataverse.org-build-develop/>
- The resulting war file is deployed to the Phoenix server: <https://build.hmdc.harvard.edu:8443/job/phoenix.dataverse.org-deploy-develop/>
- REST Assured Tests are run across the wire from the Jenkins server to the Phoenix server: <https://build.hmdc.harvard.edu:8443/job/phoenix.dataverse.org-apitest-develop/>

## How to Run the Phoenix Tests

- Take a quick look at <http://phoenix.dataverse.org> to make sure the server is up and running Dataverse. If it's down, fix it.
- Log into Jenkins and click “Build Now” at <https://build.hmdc.harvard.edu:8443/job/phoenix.dataverse.org-build-develop/>
- Wait for all three chained Jenkins jobs to complete and note if they passed or failed. If you see a failure, open a GitHub issue or at least get the attention of some developers.

## List of Tests Run Against the Phoenix Server

We haven't thought much about a good way to publicly list the “IT” classes that are executed against the phoenix server. (Currently your best bet is to look at the `Executing Maven` line at the top of the “Full Log” of “Console Output” of `phoenix.dataverse.org-apitest-develop` Jenkins job mentioned above.) We endeavor to keep the list of tests in the “all-in-one” Docker environment described above in sync with the list of tests configured in Jenkins. That is to say, refer to `run-test-suite.sh` mentioned in `conf/docker-aio/readme.txt` for the current list of IT tests that are expected to pass. Here's a dump of that file:

```
#!/bin/sh
# This is the canonical list of which "IT" tests are expected to pass.

dvurl=$1
if [ -z "$dvurl" ]; then
    dvurl="http://localhost:8084"
fi

# Please note the "dataverse.test.baseurl" is set to run for "all-in-one" Docker_
↪environment.
# TODO: Rather than hard-coding the list of "IT" classes here, add a profile to pom.
↪xml.
mvn test -Dtest=DataversesIT,DatasetsIT,SwordIT,AdminIT,BuiltinUsersIT,UsersIT,UtilIT,
↪ConfirmEmailIT,FileMetadataIT,FilesIT,SearchIT,InReviewWorkflowIT,
↪HarvestingServerIT,MoveIT,MakeDataCountApiIT,FileTypeDetectionIT,EditDDIIT,
↪ExternalToolsIT -Ddataverse.test.baseurl=$dvurl
```

## 5.8.8 Accessibility Testing

### Accessibility Policy

Dataverse aims to improve the user experience for those with disabilities, and are in the process of following the recommendations of the [Harvard University Digital Accessibility Policy](#), which use the Worldwide Web Consortium's Web Content Accessibility Guidelines version 2.1, Level AA Conformance (WCAG 2.1 Level AA) as the standard.

To report an accessibility issue with Dataverse, you can create a new issue in our GitHub repo at: <https://github.com/IQSS/dataverse/issues/>

### Accessibility Tools

Our development process will incorporate automated testing provided by tools like [SiteImprove](#) and [Accessibility Management Platform \(AMP\)](#) from Level Access, to run accessibility reports for the application.



Developers who contribute front-end UI code are responsible for understanding the requirements of this standard and the tools and methods for securing conformance with it.

There are browser developer tools such as the [Wave toolbar](#) by WebAIM (available for Chrome, Firefox) and the [Siteimprove Accessibility Checker](#) (available for Chrome, Firefox) that will generate reports for a single page. It is required that developers utilize these tools to catch any accessibility issues with pages or features that are being added to the application UI.

## 5.8.9 Future Work

We'd like to make improvements to our automated testing. See also 'this thread from our mailing list <<https://groups.google.com/forum/#!topic/dataverse-community/X8OrRWbPimA>>' \_ asking for ideas from the community, and discussion at 'this GitHub issue. <<https://github.com/IQSS/dataverse/issues/2746>>' \_

### Future Work on Unit Tests

- Review pull requests from @bencomp for ideas for approaches to testing: <https://github.com/IQSS/dataverse/pulls?q=is%3Apr+author%3Abencomp>
- Come up with a way to test commands: [http://irclog.iq.harvard.edu/dataverse/2015-11-04#i\\_26750](http://irclog.iq.harvard.edu/dataverse/2015-11-04#i_26750)
- Test EJBs using Arquillian, embedded Glassfish, or similar. @bmckinney kicked the tires on Arquillian at <https://github.com/bmckinney/bio-dataverse/commit/2f243b1db1ca704a42cd0a5de329083763b7c37a>

### Future Work on Integration Tests

- Automate testing of dataverse-client-python: <https://github.com/IQSS/dataverse-client-python/issues/10>
- Work with @leeper on testing the R client: <https://github.com/IQSS/dataverse-client-r>
- Review and attempt to implement "API Test Checklist" from @kcondon at <https://docs.google.com/document/d/199Oq1YwQ4pYCguaeW48bIN28QAitSk63NbPYxJHCCAE/edit?usp=sharing>
- Attempt to use @openscholar approach for running integration tests using Travis <https://github.com/openscholar/openscholar/blob/SCHOLAR-3.x/.travis.yml> (probably requires using Ubuntu rather than CentOS)
- Generate code coverage reports for **integration** tests: <https://github.com/pkainulainen/maven-examples/issues/3> and <http://www.petrikainulainen.net/programming/maven/creating-code-coverage-reports-for-unit-and-integration-tests-with-the-jacoco-maven-plugin/>
- Consistent logging of API Tests. Show test name at the beginning and end and status codes returned.
- expected passing and known/expected failing integration tests: <https://github.com/IQSS/dataverse/issues/4438>

### Browser-Based Testing

- Revisit Selenium/Open Sauce: <https://github.com/IQSS/dataverse/commit/8a26404> and <https://saucelabs.com/u/esodvn> and <https://saucelabs.com/u/wdjs> and <http://saucelabs.com/index.php/2013/05/a-browser-matrix-widget-for-the-open-source-community/>

### Installation Testing

- Run *vagrant up* on a server to test the installer: <http://guides.dataverse.org/en/latest/developers/tools.html#vagrant> . We haven't been able to get this working in Travis: <https://travis-ci.org/IQSS/dataverse/builds/>

96292683 . Perhaps it would be possible to use AWS as a provider from Vagrant judging from <https://circleci.com/gh/critical-alert/circleci-vagrant/6> .

- Work with @lwo to automate testing of <https://github.com/IQSS/dataverse-puppet> . Consider using Travis: <https://github.com/IQSS/dataverse-puppet/issues/10>
- Work with @donsizemore to automate testing of <https://github.com/IQSS/dataverse-ansible> with Travis or similar.

### Future Work on Load/Performance Testing

- Clean up and copy stress tests code, config, and docs into main repo from [https://github.com/IQSS/dataverse-helper-scripts/tree/master/src/stress\\_tests](https://github.com/IQSS/dataverse-helper-scripts/tree/master/src/stress_tests)
- Marcel Duran created a command-line wrapper for the WebPagetest API that can be used to test performance in your continuous integration pipeline (TAP, Jenkins, Travis-CI, etc): <https://github.com/marcelduran/webpagetest-api/wiki/Test-Specs#jenkins-integration>
- Create top-down checklist, building off the “API Test Coverage” spreadsheet at <https://github.com/IQSS/dataverse/issues/3358#issuecomment-256400776>

### Future Work on Accessibility Testing

- Using <https://github.com/IQSS/dataverse-ansible> and hooks available from accessibility testing tools, automate the running of accessibility tools on PRs so that developers will receive quicker feedback on proposed code changes that reduce the accessibility of the application.

---

Previous: *SQL Upgrade Scripts* | Next: *Writing Documentation*

## 5.9 Writing Documentation

### Contents:

- *Quick Fix*
- *Other Changes (Sphinx)*
  - *Installing Sphinx*
  - *Using Sphinx*
- *Table of Contents*
- *Images*
  - *GraphViz based images*
- *Versions*

### 5.9.1 Quick Fix

If you find a typo or a small error in the documentation you can fix it using GitHub’s online web editor. Generally speaking, we will be following <https://help.github.com/en/articles/editing-files-in-another-users-repository>

- Navigate to <https://github.com/IQSS/dataverse/tree/develop/doc/sphinx-guides/source> where you will see folders for each of the guides:
  - `admin`
  - `api`
  - `developers`
  - `installation`
  - `user`
- Find the file you want to edit under one of the folders above.
- Click the pencil icon in the upper-right corner. If this is your first contribution to Dataverse, the hover text over the pencil icon will say “Fork this project and edit this file”.
- Make changes to the file and preview them.
- In the **Commit changes** box, enter a description of the changes you have made and click **Propose file change**.
- Under the **Write** tab, delete the long welcome message and write a few words about what you fixed.
- Click **Create Pull Request**.

That’s it! Thank you for your contribution! Your pull request will be added manually to the main Dataverse project board at <https://github.com/orgs/IQSS/projects/2> and will go through code review and QA before it is merged into the “develop” branch. Along the way, developers might suggest changes or make them on your behalf. Once your pull request has been merged you will be listed as a contributor at <https://github.com/IQSS/dataverse/graphs/contributors>

Please see <https://github.com/IQSS/dataverse/pull/5857> for an example of a quick fix that was merged (the “Files changed” tab shows how a typo was fixed).

If you would like to read more about the Dataverse project’s use of GitHub, please see the *Version Control* section. For bug fixes and features we request that you create an issue before making a pull request but this is not at all necessary for quick fixes to the documentation.

## 5.9.2 Other Changes (Sphinx)

The documentation for Dataverse was written using Sphinx (<http://sphinx-doc.org/>). If you are interested in suggesting changes or updates we recommend that you create the html files using Sphinx locally and then submit a pull request through GitHub. Here are the instructions on how to proceed:

### Installing Sphinx

On a Mac:

Download the sphinx zip file from <http://sphinx-doc.org/install.html>

Unzip it somewhere. In the unzipped directory, do the following as root, (sudo -i):

```
python setup.py build python setup.py install
```

Alternative option (Mac/Unix/Windows):

Unless you already have it, install pip (<https://pip.pypa.io/en/latest/installing.html>)

```
run pip install sphinx in a terminal
```

This is all you need. You should now be able to build HTML/pdf documentation from git sources locally.

### Using Sphinx

First, you will need to make a fork of the dataverse repository in GitHub. Then, you will need to make a clone of your fork so you can manipulate the files outside GitHub.

To edit the existing documentation:

- Create a branch (refer to <http://guides.dataverse.org/en/latest/developers/version-control.html> > *Create a New Branch off the develop Branch*) to record the changes you are about to perform.
- Go to `~/dataverse/doc/sphinx-guides/source` directory inside your clone. There, you will find the `.rst` files that correspond to the guides in the dataverse page (<http://guides.dataverse.org/en/latest/>).
- Using your preferred text editor, open and edit the necessary files, or create new ones.

**NOTE:** When adding ReStructured Text (RST) [cross references](#), use the hyphen character (-) as the word separator for the cross reference label. For example, `my-reference-label` would be the preferred label for a cross reference as opposed to, for example, `my_reference_label`.

Once you are done, open a terminal and change directories to `~/dataverse/doc/sphinx-guides`. Then, run the following commands:

- `make clean`
- `make html`

After sphinx is done processing the files you should notice that the `html` folder in `~/dataverse/doc/sphinx-guides/build` directory has been updated. You can click on the files in the `html` folder to preview the changes.

Now you can make a commit with the changes to your own fork in GitHub and submit a pull request to the original (upstream) dataverse repository.

### 5.9.3 Table of Contents

Every non-index page should use the following code to display a table of contents of internal sub-headings:

```
.. contents:: |toctitle|
   :local:
```

This code should be placed below any introductory text/images and directly above the first subheading, much like a Wikipedia page.

### 5.9.4 Images

A good documentation is just like a website enhanced and upgraded by adding high quality and self-explanatory images. Often images depict a lot of written text in a simple manner. Within our Sphinx docs, you can add them in two ways: a) add a PNG image directly and include or b) use inline description languages like GraphViz (current only option).

While PNGs in the git repo can be linked directly via URL, Sphinx-generated images do not need a manual step and might provide higher visual quality. Especially in terms of quality of content, generated images can be extended and improved by a textbased and reviewable commit, without needing raw data or source files and no diff around.

#### GraphViz based images

In some parts of the documentation, graphs are rendered as images via Sphinx GraphViz extension.

This requires [GraphViz](#) installed and either `dot` on the path or [adding options to the make call](#).

This has been tested and works on Mac, Linux, and Windows. If you have not properly configured GraphViz, then the worst thing that might happen is a warning and missing images in your local documentation build.

## 5.9.5 Versions

For installations hosting their own copies of the guides, note that as each version of Dataverse is released, there is an updated version of the guides released with it. Google and other search engines index all versions, which may confuse users who discover your guides in the search results as to which version they should be looking at. When learning about your installation from the search results, it is best to be viewing the *latest* version.

In order to make it clear to the crawlers that we only want the latest version discoverable in their search results, we suggest adding this to your `robots.txt` file:

```
User-agent: *
Allow: /en/latest/
Disallow: /en/
```

Previous: *Testing* | Next: *Dependency Management*

## 5.10 Dependency Management

### Contents:

- *Terms*
- *Direct dependencies*
- *Transitive dependencies*
  - *Managing transitive dependencies in pom.xml*
- *Helpful tools*
- *Repositories*

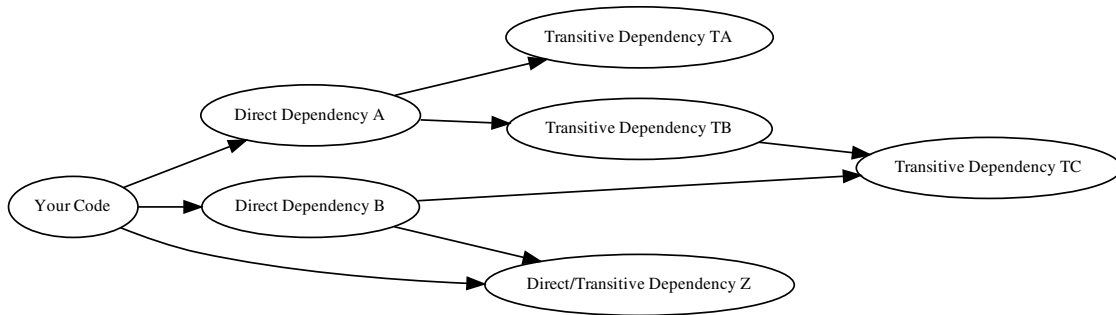
Dataverse is (currently) a Java EE 7 based application, that uses a lot of additional libraries for special purposes. This includes features like support for SWORD-API, S3 storage and many others.

Besides the code that glues together the single pieces, any developer needs to describe used dependencies for the Maven-based build system. As is familiar to any Maven user, this happens inside the “Project Object Model” (POM) living in `pom.xml` at the root of the project repository. Recursive and convergent dependency resolution makes dependency management with Maven very easy. But sometimes, in projects with many complex dependencies like Dataverse, you have to help Maven make the right choices.

### 5.10.1 Terms

As a developer, you should familiarize yourself with the following terms:

- **Direct dependencies:** things *you use* yourself in your own code for Dataverse.
- **Transitive dependencies:** things *others use* for things you use, pulled in recursively. See also: [Maven docs](#).



## 5.10.2 Direct dependencies

Within the POM, any direct dependencies reside within the `<dependencies>` tag:

```

<dependencies>
  <dependency>
    <groupId>org.example</groupId>
    <artifactId>example</artifactId>
    <version>1.1.0</version>
    <scope>compile</scope>
  </dependency>
</dependencies>

```

Anytime you add a `<dependency>`, Maven will try to fetch it from defined/configured repositories and use it within the build lifecycle. You have to define a `<version>`, but `<scope>` is optional for `compile`. (See [Maven docs: Dep. Scope](#))

During fetching, Maven will analyse all transitive dependencies (see graph above) and, if necessary, fetch those, too. Everything downloaded once is cached locally by default, so nothing needs to be fetched again and again, as long as the dependency definition does not change.

### Rules to follow:

1. You should only use direct dependencies for **things you are actually using** in your code.
2. **Clean up** direct dependencies no longer in use. It will bloat the deployment package otherwise!
3. Care about the **scope**. Do not include “testing only” dependencies in the package - it will hurt you in IDEs and bloat things.<sup>1</sup>
4. Avoid using different dependencies for the **same purpose**, e. g. different JSON parsing libraries.
5. Refactor your code to **use Java EE** standards as much as possible.
6. When you rely on big SDKs or similar big cool stuff, try to **include the smallest portion possible**. Complete SDK bundles are typically heavyweight and most of the time unnecessary.
7. **Don’t include transitive dependencies.**<sup>2</sup>
  - Exception: if you are relying on it in your code (see Z in the graph above), you must declare it. See below for proper handling in these (rare) cases.

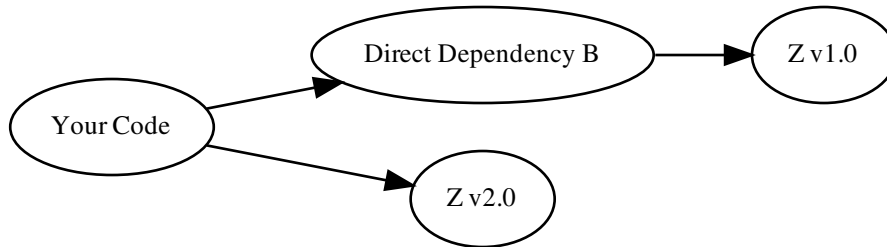
<sup>1</sup> Modern IDEs import your Maven POM and offer import autocompletion for classes based on direct dependencies in the model. You might end up using legacy or repackaged classes because of a wrong scope.

<sup>2</sup> This is going to bite back in modern IDEs when importing classes from transitive dependencies by “autocompletion accident”.

### 5.10.3 Transitive dependencies

Maven is comfortable for developers; it handles recursive resolution, downloading, and adding “dependencies of dependencies”. However, as life is a box of chocolates, you might find yourself in *version conflict hell* sooner than later without even knowing, but experiencing unintended side effects.

When you look at the graph above, imagine *B* and *TB* rely on different *versions* of *TC*. How does Maven decide which version it will include? Easy: the dependent version of the nearest version wins:



In this case, version “2.0” will be included. If you know something about semantic versioning, a red alert should ring in your mind right now. How do we know that *B* is compatible with *Z v2.0* when depending on *Z v1.0*?

Another scenario getting us in trouble: indirect use of transitive dependencies. Imagine the following: we rely on *Z* in our code, but do not include a direct dependency for it within the POM. Now *B* is updated and removed its dependency on *Z*. You definitely don’t want to head down that road.

**Follow the rules to be safe:**

1. Do **not use transitive deps implicit**: add a direct dependency for transitive deps you re-use in your code.
2. On every build check that no implicit usage was added by accident.
3. **Explicitly declare versions** of transitive dependencies in use by multiple direct dependencies.
4. On every build check that there are no convergence problems hiding in the shadows.
5. **Do special tests** on every build to verify these explicit combinations work.

#### Managing transitive dependencies in `pom.xml`

Maven can manage versions of transitive dependencies in four ways:

1. Make a transitive-only dependency not used in your code a direct one and add a `<version>` tag. Typically a bad idea, don’t do that.
2. Use `<optional>` or `<exclusion>` tags on direct dependencies that request the transitive dependency. *Last resort*, you really should avoid this. Not explained or used here. [See Maven docs](#).
3. Explicitly declare the transitive dependency in `<dependencyManagement>` and add a `<version>` tag.
4. For more complex transitive dependencies, reuse a “Bill of Materials” (BOM) within `<dependencyManagement>` and add a `<version>` tag. Many bigger and standard use projects provide those, making the POM much less bloated compared to adding every bit yourself.

A reduced example, only showing bits relevant to the above cases and usage of an explicit transitive dep directly:

```

1 <properties>
2   <aws.version>1.11.172</aws.version>
3   <!-- We need to ensure that our chosen version is compatible with every_
↳ dependency relying on it.
4     This is manual work and needs testing, but a good investment in stability_
↳ and up-to-date dependencies. -->
5   <jackson.version>2.9.6</jackson.version>
6   <joda.version>2.10.1</joda.version>
7 </properties>
8
9 <!-- Transitive dependencies, bigger library "bill of materials" (BOM) and
10 versions of dependencies used both directly and transitive are managed here. -->
11 <dependencyManagement>
12   <dependencies>
13     <!-- First example for case 4. Only one part of the SDK (S3) is used and_
↳ transitive deps
14       of that are again managed by the upstream BOM. -->
15     <dependency>
16       <groupId>com.amazonaws</groupId>
17       <artifactId>aws-java-sdk-bom</artifactId>
18       <version>${aws.version}</version>
19       <type>pom</type>
20       <scope>import</scope>
21     </dependency>
22     <!-- Second example for case 4 and an example for explicit direct usage of a_
↳ transitive dependency.
23       Jackson is used by AWS SDK and others, but we also use it in Dataverse. -
↳ ->
24     <dependency>
25       <groupId>com.fasterxml.jackson</groupId>
26       <artifactId>jackson-bom</artifactId>
27       <version>${jackson.version}</version>
28       <scope>import</scope>
29       <type>pom</type>
30     </dependency>
31     <!-- Example for case 3. Joda is not used in Dataverse (as of writing this). -
↳ ->
32     <dependency>
33       <groupId>joda-time</groupId>
34       <artifactId>joda-time</artifactId>
35       <version>${joda.version}</version>
36     </dependency>
37   </dependencies>
38 </dependencyManagement>
39
40 <!-- Declare any DIRECT dependencies here.
41   In case the dependency is both transitive and direct (e. g. some common lib for_
↳ logging),
42   manage the version above and add the direct dependency here WITHOUT version tag,
↳ too.
43 -->
44 <dependencies>
45   <dependency>
46     <groupId>com.amazonaws</groupId>
47     <artifactId>aws-java-sdk-s3</artifactId>
48     <!-- no version here as managed by BOM above! -->
49   </dependency>
50   <!-- Should be refactored and removed once on Java EE 8 -->

```



```

51 <dependency>
52 <groupId>com.fasterxml.jackson.core</groupId>
53 <artifactId>jackson-core</artifactId>
54 <!-- no version here as managed above! -->
55 </dependency>
56 <!-- Should be refactored and removed once on Java EE 8 -->
57 <dependency>
58 <groupId>com.fasterxml.jackson.core</groupId>
59 <artifactId>jackson-databind</artifactId>
60 <!-- no version here as managed above! -->
61 </dependency>
62 </dependencies>

```

### 5.10.4 Helpful tools

Maven provides some plugins that are of great help to detect possible conflicts and implicit usage.

For *implicit usage detection*, use `mvn dependency:analyze`. Examine the output with great care. Sometimes you will see implicit usages that do no harm, especially if you are using bigger SDKs having some kind of *core* package. This will also report on any direct dependency which is not in use and can be removed from the POM. Again, do this with great caution and double check.

If you want to see the dependencies both direct and transitive in a *dependency tree format*, use `mvn dependency:tree`.

This will however not help you with detecting possible version conflicts. For this you need to use the [Enforcer Plugin](#) with its built in [dependency convergence rule](#).

### 5.10.5 Repositories

Maven receives all dependencies from *repositories*. Those can be public like [Maven Central](#) and others, but you can also use a private repository on premises or in the cloud. Last but not least, you can use local repositories, which can live next to your application code (see `local_lib` dir within Dataverse codebase).

Repositories are defined within the Dataverse POM like this:

```

<repositories>
  <repository>
    <id>central-repo</id>
    <name>Central Repository</name>
    <url>http://repo1.maven.org/maven2</url>
    <layout>default</layout>
  </repository>
  <repository>
    <id>prime-repo</id>
    <name>PrimeFaces Maven Repository</name>
    <url>http://repository.primefaces.org</url>
    <layout>default</layout>
  </repository>
  <repository>
    <id>dvn.private</id>
    <name>Local repository for hosting jars not available from network_
↪ repositories.</name>
    <url>file://${project.basedir}/local_lib</url>
  </repository>
</repositories>

```

You can also add repositories to your local Maven settings, see [docs](#).

Typically you will skip the addition of the central repository, but adding it to the POM has the benefit that dependencies are first looked up there (which in theory can speed up downloads). You should keep in mind that repositories are used in the order they appear.

---

---

Previous: [Writing Documentation](#) | Next: [Debugging](#)

## 5.11 Debugging

### Contents:

- [Logging](#)

### 5.11.1 Logging

By default, Glassfish logs at the “INFO” level but logging can be increased to “FINE” on the fly with (for example) `./asadmin set-log-levels edu.harvard.iq.dataverse.api.Datasets=FINE`. Running `./asadmin list-log-levels` will show the current logging levels.

---

Previous: [Writing Documentation](#) | Next: [Coding Style](#)

## 5.12 Coding Style

Like all development teams, the Dataverse developers at IQSS have their habits and styles when it comes to writing code. Let’s attempt to get on the same page. :)

### Contents:

- [Java](#)
  - [Formatting Code](#)
    - \* [Tabs vs. Spaces](#)
    - \* [Braces Placement](#)
    - \* [Format Code You Changed with Netbeans](#)
    - \* [Checking Your Formatting With Checkstyle](#)
  - [Logging](#)
  - [Avoid Hard-Coding Strings \(Use Constants\)](#)
  - [Avoid Hard-Coding User-Facing Messaging in English](#)

- *Type Safety*
- *Bash*
  - *Formatting Code*
    - \* *Tabs vs. Spaces*
- *Bike Shedding*

## 5.12.1 Java

### Formatting Code

#### Tabs vs. Spaces

Don't use tabs. Use 4 spaces.

#### Braces Placement

Place curly braces according to the style below, which is an example you can see from Netbeans.

```
public class ClassA {

    private String letters[] = new String[]{"A", "B"};

    public int meth(String text, int number) {
        BinaryOperator plus = (a, b) -> {
            return a + b;
        };
        if (text != null) {
            try {
                meth("Some text", text.length());
            } catch (Throwable t) {
            } finally {
            }
        } else if (number >= 0) {
            text = number == 0 ? "empty" : "nonempty";
        }
        do {
            number = number + 1;
        } while (number < 2);
        for (int i = 1; i < 100; i++) {
            number = number + i;
        }
        while (number > 0) {
            number--;
        }
    }
}
```

### Format Code You Changed with Netbeans

As you probably gathered from the *Development Environment* section, IQSS has standardized on Netbeans. It is much appreciated when you format your code (but only the code you touched!) using the out-of-the-box Netbeans configuration. If you have created an entirely new Java class, you can just click Source -> Format. If you are adjusting code in an existing class, highlight the code you changed and then click Source -> Format. Keeping the “diff” in your pull requests small makes them easier to code review.

### Checking Your Formatting With Checkstyle

The easiest way to adopt Dataverse coding style is to use Netbeans as your IDE, avoid change the default Netbeans formatting settings, and only reformat code you’ve changed, as described above.

If you do not use Netbeans, you are encouraged to check the formatting of your code using Checkstyle.

To check the entire project:

```
mvn checkstyle:checkstyle
```

To check a single file:

```
mvn checkstyle:checkstyle -Dcheckstyle.includes=**\*/SystemConfig*.java
```

### Logging

We have adopted a pattern where the top of every class file has a line like this:

```
private static final Logger logger = Logger.getLogger(DatasetUtil.class.  
↳getCanonicalName());
```

Use this logger field with varying levels such as `fine` or `info` like this:

```
logger.fine("will get thumbnail from dataset logo");
```

Generally speaking you should use `fine` for everything that you don’t want to show up in Glassfish’s `server.log` file by default. If you use a higher level such as `info` for common operations, you will probably hear complaints that your code is too “chatty” in the logs. These logging levels can be controlled at runtime both on your development machine and in production as explained in the *Debugging* section.

When adding logging, do not simply add `System.out.println()` lines because the logging level cannot be controlled.

### Avoid Hard-Coding Strings (Use Constants)

Special strings should be defined as public constants. For example, `DatasetFieldConstant.java` contains a field for “title” and it’s used in many places in the code (try “Find Usages” in Netbeans). This is better than writing the string “title” in all those places.

### Avoid Hard-Coding User-Facing Messaging in English

There is an ongoing effort to translate Dataverse into various languages. Look for “lang” or “languages” in the *Configuration* section of the Installation Guide for details if you’d like to help or play around with this feature.

The translation effort is hampered if you hard code user-facing messages in English in the Java code. Put English strings in `Bundle.properties` and use `BundleUtil` to pull them out. This is especially important for messages

that appear in the UI. We are aware that the API has many, many hard coded English strings in it. If you touch a method in the API and notice English strings, you are strongly encouraged to use that opportunity to move the English to `Bundle.properties`.

## Type Safety

If you just downloaded Netbeans and are using the out-of-the-box settings, you should be in pretty good shape. Unfortunately, the default configuration of Netbeans doesn't warn you about type-safety problems you may be inadvertently introducing into the code. To see these warnings, click Netbeans -> Preferences -> Editor -> Hints and check the following:

- “Raw Types” under “Standard Javac Warnings”

If you know of a way to easily share Netbeans configuration across a team, please get in touch.

## 5.12.2 Bash

Generally, Google's Shell Style Guide at <https://google.github.io/styleguide/shell.xml> seems to have good advice.

### Formatting Code

#### Tabs vs. Spaces

Don't use tabs. Use 2 spaces.

shfmt from <https://github.com/mvdan/sh> seems like a decent way to enforce indentation of two spaces (i.e. `shfmt -i 2 -w path/to/script.sh`) but be aware that it makes other changes.

## 5.12.3 Bike Shedding

What color should the `bike shed` be? :)

Come debate with us about coding style in this Google doc that has public comments enabled: <https://docs.google.com/document/d/1KTd3FpM1BI3HIBofaZjMmBiQEJtFf11jiiGpQeJzy7A/edit?usp=sharing>

---

Previous: *Debugging* | Next: *Deployment*

## 5.13 Deployment

Developers often only deploy Dataverse to their *Development Environment* but it can be useful to deploy Dataverse to cloud services such as Amazon Web Services (AWS).

### Contents:

- *Deploying Dataverse to Amazon Web Services (AWS)*
  - *Install AWS CLI*
  - \* *Troubleshooting “aws: command not found”*

- *Configure AWS CLI*
- *Configure Ansible File (Optional)*
- *Download and Run the “Create Instance” Script*
- *Caveats*

### 5.13.1 Deploying Dataverse to Amazon Web Services (AWS)

We have written scripts to deploy Dataverse to Amazon Web Services (AWS) but they require some setup.

#### Install AWS CLI

First, you need to have AWS Command Line Interface (AWS CLI) installed, which is called `aws` in your terminal. Launching your terminal and running the following command to print out the version of AWS CLI will tell you if it is installed or not:

```
aws --version
```

If you have not yet installed AWS CLI you should install it by following the instructions at <https://docs.aws.amazon.com/cli/latest/userguide/installing.html>

Afterwards, you should re-run the “version” command above to verify that AWS CLI has been properly installed. If “version” still doesn’t work, read on for troubleshooting advice. If “version” works, you can skip down to the “Configure AWS CLI” step.

#### Troubleshooting “aws: command not found”

Please note that as of this writing the AWS docs are not especially clear about how to fix errors such as `aws: command not found`. If the AWS CLI cannot be found after you followed the AWS installation docs, it is very likely that the `aws` program is not in your `$PATH`. `$PATH` is an “environment variable” that tells your shell (usually Bash) where to look for programs.

To see what `$PATH` is set to, run the following command:

```
echo $PATH
```

On Mac, to update your `$PATH` to include the location where the current AWS docs install AWS CLI on the version of Python included with your Mac, run the following command:

```
export PATH=$PATH:$HOME/Library/Python/2.7/bin
```

After all this, you can try the “version” command again.

Note that it’s possible to add an `export` line like the one above to your `~/.bash_profile` file so you don’t have to run it yourself when you open a new terminal.

#### Configure AWS CLI

Next you need to configure AWS CLI.

Create a `.aws` directory in your home directory (which is called `~`) like this:

```
mkdir ~/.aws
```

We will be creating two plain text files in the `.aws` directory and it is important that these files do not end in `.txt` or any other extension. After creating the files, you can verify their names with the following command:

```
ls ~/.aws
```

Create a plain text file at `~/.aws/config` with the following content:

```
[default]
region = us-east-1
```

Please note that at this time the region must be set to `us-east-1` but in the future we could improve our scripts to support other regions.

Create a plain text file at `~/.aws/credentials` with the following content:

```
[default]
aws_access_key_id = XXXXXXXXXXXXXXXXXXXX
aws_secret_access_key = XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Then update the file and replace the values for `aws_access_key_id` and `aws_secret_access_key` with your actual credentials by following the instructions at <https://aws.amazon.com/blogs/security/wheres-my-secret-access-key/>

If you are having trouble configuring the files manually as described above, see <https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html> which documents the `aws configure` command.

### Configure Ansible File (Optional)

In order to configure Dataverse settings such as the password of the `dataverseAdmin` user, download <https://raw.githubusercontent.com/IQSS/dataverse-ansible/master/defaults/main.yml> and edit the file to your liking.

You can skip this step if you're fine with the values in the `main.yml` file in the link above.

### Download and Run the "Create Instance" Script

Once you have done the configuration above, you are ready to try running the `ec2-create-instance.sh` script to spin up Dataverse in AWS.

Download `ec2-create-instance.sh` and put it somewhere reasonable. For the purpose of these instructions we'll assume it's in the `Downloads` directory in your home directory.

`ec2-create-instance` accepts a number few command-line switches:

- `-r`: GitHub Repository URL (defaults to <https://github.com/IQSS/dataverse.git>)
- `-b`: branch to build (defaults to `develop`)
- `-p`: pemfile directory (defaults to `$HOME`)
- `-g`: Ansible GroupVars file (if you wish to override role defaults)

```
bash ~/Downloads/ec2-create-instance.sh -b develop -r https://github.com/scholarsportal/dataverse.git -g main.yml
```

Now you will need to wait around 15 minutes until the deployment is finished. Eventually, the output should tell you how to access the installation of Dataverse in a web browser or via ssh. It will also provide instructions on how to delete the instance when you are finished with it. Please be aware that AWS charges per minute for a running instance. You can also delete your instance from <https://console.aws.amazon.com/console/home?region=us-east-1>.

## Caveats

Please note that while the script should work fine on newish branches, older branches that have different dependencies such as an older version of Solr may not produce a working Dataverse installation. Your mileage may vary.

---

Previous: *Coding Style* | Next: *Docker, Kubernetes, and OpenShift*

## 5.14 Docker, Kubernetes, and OpenShift

Dataverse is exploring the use of Docker, Kubernetes, OpenShift and other container-related technologies.

### Contents:

- *OpenShift*
  - *Install Minishift*
  - *Start Minishift*
  - *Make the OpenShift Client Binary (oc) Executable*
  - *Log in to Minishift from the Command Line*
  - *Create a Minishift Project*
  - *Create a Dataverse App within the Minishift Project*
  - *Log into Minishift and Visit Dataverse in your Browser*
  - *Troubleshooting*
    - \* *Check Status of Dataverse Deployment to Minishift*
    - \* *Review Logs of Dataverse Deployment to Minishift*
    - \* *Get a Shell (ssh/rsh) on Containers Deployed to Minishift*
  - *Cleaning up*
  - *Making Changes*
    - \* *Making Changes to Docker Images*
    - \* *Making Changes to the OpenShift Config*
    - \* *Making Changes to the PostgreSQL Database from the Glassfish Pod*
  - *Scaling Dataverse by Increasing Replicas in a StatefulSet*
  - *Configuring Persistent Volumes and Solr master node recovery*
  - *Running Containers to Run as Root in Minishift*
  - *Minishift Resources*
- *Docker*
  - *Installing Docker*
  - *All In One Docker Images for Testing*
  - *Future production use on Minishift/OpenShift/Kubernetes*



\* *Known Issues with Dataverse Images on Docker Hub*

### 5.14.1 OpenShift

From the Dataverse perspective, we are in the business of providing a “template” for OpenShift that describes how the various components we build our application on (Glassfish, PostgreSQL, Solr, the Dataverse war file itself, etc.) work together. We publish Docker images to DockerHub at <https://hub.docker.com/u/iqss/> that are used in this OpenShift template.

Dataverse’s (light) use of Docker is documented below in a separate section. We actually started with Docker in the context of OpenShift, which is why OpenShift is listed first but we can imagine rearranging this in the future.

The OpenShift template for Dataverse can be found at `conf/openshift/openshift.json` and if you need to hack on the template or related files under `conf/docker` it is recommended that you iterate on them using Minishift.

The instructions below will walk you through spinning up Dataverse within Minishift. It is recommended that you do this on the “develop” branch to make sure everything is working before changing anything.

#### Install Minishift

Minishift requires a hypervisor and since we already use VirtualBox for Vagrant, you should install VirtualBox from <http://virtualbox.org>.

Download the Minishift tarball from <https://docs.openshift.org/latest/minishift/getting-started/installing.html> and put the `minishift` binary in `/usr/local/bin` or somewhere in your `$PATH`. This assumes Mac or Linux. These instructions were last tested on version `v1.14.0+1ec5877` of Minishift.

At this point, you might want to consider going through the Minishift quickstart to get oriented: <https://docs.openshift.org/latest/minishift/getting-started/quickstart.html>

#### Start Minishift

```
minishift start --vm-driver=virtualbox --memory=8GB
```

#### Make the OpenShift Client Binary (oc) Executable

```
eval $(minishift oc-env)
```

#### Log in to Minishift from the Command Line

Note that if you just installed and started Minishift, you are probably logged in already. This `oc login` step is included in case you aren’t logged in anymore.

```
oc login --username developer --password=whatever
```

Use “developer” as the username and a couple characters as the password.

### Create a Minishift Project

Calling the project “project1” is fairly arbitrary. We’ll probably want to revisit this name in the future. A project is necessary in order to create an OpenShift app.

```
oc new-project project1
```

Note that `oc projects` will return a list of projects.

### Create a Dataverse App within the Minishift Project

The following command operates on the `conf/openshift/openshift.json` file that resides in the main Dataverse git repo. It will download images from Docker Hub and use them to spin up Dataverse within Minishift/OpenShift. Later we will cover how to make changes to the images on Docker Hub.

```
oc new-app conf/openshift/openshift.json
```

### Log into Minishift and Visit Dataverse in your Browser

After running the `oc new-app` command above, deployment of Dataverse within Minishift/OpenShift will begin. You should log into the OpenShift web interface to check on the status of the deployment. If you just created the Minishift VM with the `minishift start` command above, the `oc new-app` step is expected to take a while because the images need to be downloaded from Docker Hub. Also, the installation of Dataverse takes a while.

Typing `minishift console` should open the OpenShift web interface in your browser. The IP address might not be “192.168.99.100” but it’s used below as an example.

- `https://192.168.99.100:8443` (or URL from `minishift console`)
- username: developer
- password: <any password>

In the OpenShift web interface you should see a link that looks something like `http://dataverse-project1.192.168.99.100.nip.io` but the IP address will vary and will match the output of `minishift ip`. Eventually, after deployment is complete, the Dataverse web interface will appear at this URL and you will be able to log in with the username “dataverseAdmin” and the password “admin”.

Another way to verify that Dataverse has been successfully deployed is to make sure that the Dataverse “info” API endpoint returns a version (note that `minishift ip` is used because the IP address will vary):

```
curl http://dataverse-project1.`minishift ip`.nip.io/api/info/version
```

From the perspective of OpenShift and the `openshift.json` config file, the HTTP link to Dataverse is called a route. See also documentation for `oc expose`.

### Troubleshooting

Here are some tips on troubleshooting your deployment of Dataverse to Minishift.

### Check Status of Dataverse Deployment to Minishift

```
oc status
```

Once images have been downloaded from Docker Hub, the output below will change from `Pulling` to `Pulled`.

```
oc get events | grep Pull
```

This is a deep dive:

```
oc get all
```

## Review Logs of Dataverse Deployment to Minishift

Logs are provided in the web interface to each of the deployment configurations. The URLs should be something like this (but the IP address) will vary and you should click “View Log”. The installation of Dataverse is done within the one Glassfish deployment configuration:

- <https://192.168.99.100:8443/console/project/project1/browse/dc/dataverse-glassfish>
- <https://192.168.99.100:8443/console/project/project1/browse/dc/dataverse-postgresql>
- <https://192.168.99.100:8443/console/project/project1/browse/dc/dataverse-solr>

You can also see logs from each of the components (Glassfish, PostgreSQL, and Solr) from the command line with `oc logs` like this (just change the `grep` at the end):

```
oc logs $(oc get po -o json | jq '.items[] | select(.kind=="Pod").metadata.name' -r | grep glassfish)
```

## Get a Shell (ssh/rsh) on Containers Deployed to Minishift

You can get a shell on any of the containers for each of the components (Glassfish, PostgreSQL, and Solr) with `oc rc` (just change the `grep` at the end):

```
oc rsh $(oc get po -o json | jq '.items[] | select(.kind=="Pod").metadata.name' -r | grep glassfish)
```

From the `rsh` prompt of the Glassfish container you could run something like the following to make sure that Dataverse is running on port 8080:

```
curl http://localhost:8080/api/info/version
```

## Cleaning up

If you simply wanted to try out Dataverse on Minishift and want to clean up, you can run `oc delete project project1` to delete the project or `minishift stop` and `minishift delete` to delete the entire Minishift VM and all the Docker containers inside it.

## Making Changes

### Making Changes to Docker Images

If you’re interested in using Minishift for development and want to change the Dataverse code, you will need to get set up to create Docker images based on your changes and make them available within Minishift.

It is recommended to add experimental images to Minishift’s internal registry. Note that despite what <https://docs.openshift.org/latest/minishift/openshift/openshift-docker-registry.html> says you will not use `docker push` because we have seen “unauthorized: authentication required” when trying to push to it as reported at <https://github.com/minishift/minishift/issues/817>. Rather you will run `docker build` and run `docker images` to see that your newly build images are listed in Minishift’s internal registry.

First, set the Docker environment variables so that `docker build` and `docker images` refer to the internal Minishift registry rather than your normal Docker setup:

```
eval $(minishift docker-env)
```

When you're ready to build, change to the right directory:

```
cd conf/docker
```

And then run the build script in “internal” mode:

```
./build.sh internal
```

Note that `conf/openshift/openshift.json` must not have `imagePullPolicy` set to `Always` or it will pull from “iqss” on Docker Hub. Changing it to `IfNotPresent` allow Minishift to use the images shown from `docker images` rather than the ones on Docker Hub.

Using Minishift for day to day Dataverse development might be something we want to investigate in the future. These blog posts talk about developing Java applications using Minishift/OpenShift:

- <https://blog.openshift.com/fast-iterative-java-development-on-openshift-kubernetes-using-rsync/>
- <https://blog.openshift.com/debugging-java-applications-on-openshift-kubernetes/>

### Making Changes to the OpenShift Config

If you are interested in changing the OpenShift config file for Dataverse at `conf/openshift/openshift.json` note that in many cases once you have Dataverse running in Minishift you can use `oc process` and `oc apply` like this (but please note that some errors and warnings are expected):

```
oc process -f conf/openshift/openshift.json | oc apply -f -
```

The slower way to iterate on the `openshift.json` file is to delete the project and re-create it.

### Making Changes to the PostgreSQL Database from the Glassfish Pod

You can access and modify the PostgreSQL database via an interactive terminal called `psql`.

To log in to `psql` from the command line of the Glassfish pod, type the following command:

```
PGPASSWORD=$POSTGRES_PASSWORD; export PGPASSWORD; /usr/bin/psql  
-h $POSTGRES_SERVER.$POSTGRES_SERVICE_HOST -U $POSTGRES_USER -d  
$POSTGRES_DATABASE
```

To log in as an admin, type this command instead:

```
PGPASSWORD=$POSTGRES_ADMIN_PASSWORD; export PGPASSWORD; /usr/bin/psql -h  
$POSTGRES_SERVER.$POSTGRES_SERVICE_HOST -U postgres -d $POSTGRES_DATABASE
```

### Scaling Dataverse by Increasing Replicas in a StatefulSet

Glassfish, Solr and PostgreSQL Pods are in a “StatefulSet” which is a concept from OpenShift and Kubernetes that you can read about at <https://kubernetes.io/docs/concepts/workloads/controllers/statefulset/>

As of this writing, the `openshift.json` file has a single “replica” for each of these two stateful sets. It’s possible to increase the number of replicas from 1 to 3, for example, with this command:

```
oc scale statefulset/dataverse-glassfish --replicas=3
```

The command above should result in two additional Glassfish pods being spun up. The name of the pods is significant and there is special logic in the “zeroth” pod (“`dataverse-glassfish-0`” and “`dataverse-postgresql-0`”). For example, only “`dataverse-glassfish-0`” makes itself the dedicated timer server as explained in *Dataverse Application Timers* section of the Admin Guide. “`dataverse-glassfish-1`” and other higher number pods will not be configured as a timer server.

Once you have multiple Glassfish servers you may notice bugs that will require additional configuration to fix. One such bug has to do with Dataverse logos which are stored at `/usr/local/glassfish4/glassfish/domains/domain1/docroot/logos` on each of the Glassfish servers. This means that the logo will look fine when you just uploaded it because you're on the server with the logo on the local file system but when you visit that dataverse in the future and you're on a different Glassfish server, you will see a broken image. (You can find some discussion of this logo bug at <https://github.com/IQSS/dataverse-aws/issues/10> and <http://irclog.iq.harvard.edu/dataverse/2016-10-21>.) This is all "advanced" installation territory (see the *Advanced Installation* section of the Installation Guide) and OpenShift might be a good environment in which to work on some of these bugs.

Multiple PostgreSQL servers are possible within the OpenShift environment as well and have been set up with some amount of replication. "dataverse-postgresql-0" is the master and non-zero pods are the slaves. We have just scratched the surface of this configuration but replication from master to slave seems to be working. Future work could include failover and making Dataverse smarter about utilizing multiple PostgreSQL servers for reads. Right now we assume Dataverse is only being used with a single PostgreSQL server and that it's the master.

Solr supports index distribution and replication for scaling. For OpenShift use, we choose replication. It's possible to scale up Solr using the method similar to Glassfish, as mentioned above. In OpenShift, the first Solr pod, `dataverse-solr-0`, will be the master node, and the rest will be slave nodes.

### Configuring Persistent Volumes and Solr master node recovery

Solr requires backing up the search index to persistent storage. For our proof of concept, we configure a `hostPath`, which allows Solr containers to access the hosts' file system, for our Solr containers backups. To read more about OpenShift/Kubernetes' persistent volumes, please visit: <https://kubernetes.io/docs/concepts/storage/persistent-volumes>

To allow containers to use a host's storage, we need to allow access to that directory first. In this example, we expose `/tmp/share` to the containers:

```
# mkdir /tmp/share
# chcon -R -t svirt_sandbox_file_t
# chgrp root -R /tmp/share
# oc login -u system:admin
# oc edit scc restricted # Update allowHostDirVolumePlugin to true and runAsUser_
→type to RunAsAny
```

To add a persistent volume and persistent volume claim, in `conf/docker/openshift/openshift.json`, add the following to objects in `openshift.json`. Here, we are using `hostPath` for development purposes. Since OpenShift supports many types of cluster storages, if the administrator wishes to use any cluster storage like EBS, Google Cloud Storage, etc, they would have to use a different type of Persistent Storage:

```
{
  "kind" : "PersistentVolume",
  "apiVersion" : "v1",
  "metadata":{
    "name" : "solr-index-backup",
    "labels":{
      "name" : "solr-index-backup",
      "type" : "local"
    }
  },
  "spec":{
    "capacity":{
      "storage" : "8Gi"
    },
    "accessModes":[
      "ReadWriteMany", "ReadWriteOnce", "ReadOnlyMany"
    ]
  }
}
```

```
    ],
    "hostPath": {
      "path" : "/tmp/share"
    }
  },
  {
    "kind" : "PersistentVolumeClaim",
    "apiVersion": "v1",
    "metadata": {
      "name": "solr-claim"
    },
    "spec": {
      "accessModes": [
        "ReadWriteMany", "ReadWriteOnce", "ReadOnlyMany"
      ],
      "resources": {
        "requests": {
          "storage": "3Gi"
        }
      },
      "selector": {
        "matchLabels": {
          "name" : "solr-index-backup",
          "type" : "local"
        }
      }
    }
  }
}
```

To make solr container mount the hostPath, add the following part under .spec.spec (for Solr StatefulSet):

```
{
  "kind": "StatefulSet",
  "apiVersion": "apps/v1beta1",
  "metadata": {
    "name": "dataverse-solr",
    ....
  },
  "spec": {
    "serviceName" : "dataverse-solr-service",
    ....

    "spec": {
      "volumes": [
        {
          "name": "solr-index-backup",
          "persistentVolumeClaim": {
            "claimName": "solr-claim"
          }
        }
      ],
      ....

      "containers": [
        ....

        "volumeMounts": [
          {
```

```

    "mountPath" : "/var/share",
    "name" : "solr-index-backup"
  }

```

Solr is now ready for backup and recovery. In order to backup:

```

oc rsh dataverse-solr-0
curl 'http://localhost:8983/solr/collection1/replication?command=backup&location=/var/
↪share'

```

In `solr` `entrypoint.sh`, it's configured so that if `dataverse-solr-0` failed, it will get the latest version of the index in the backup and restore. All backups are stored in `/tmp/share` in the host, or `/home/share` in `solr` containers.

## Running Containers to Run as Root in Minishift

It is **not** recommended to run containers as root in Minishift because for security reasons OpenShift doesn't support running containers as root. However, it's good to know how to allow containers to run as root in case you need to work on a Docker image to make it run as non-root.

For more information on improving Docker images to run as non-root, see "Support Arbitrary User IDs" at [https://docs.openshift.org/latest/creating\\_images/guidelines.html#openshift-origin-specific-guidelines](https://docs.openshift.org/latest/creating_images/guidelines.html#openshift-origin-specific-guidelines)

Let's say you have a container that you suspect works fine when it runs as root. You want to see it working as-is before you start hacking on the Dockerfile and `entrypoint` file. You can configure Minishift to allow containers to run as root with this command:

```
oc adm policy add-scc-to-user anyuid -z default --as system:admin
```

Once you are done testing you can revert Minishift back to not allowing containers to run as root with this command:

```
oc adm policy remove-scc-from-user anyuid -z default --as system:admin
```

## Minishift Resources

The following resources might be helpful.

- <https://blog.openshift.com/part-1-from-app-to-openshift-runtimes-and-templates/>
- <https://blog.openshift.com/part-2-creating-a-template-a-technical-walkthrough/>
- [https://docs.openshift.com/enterprise/3.0/architecture/core\\_concepts/templates.html](https://docs.openshift.com/enterprise/3.0/architecture/core_concepts/templates.html)

## 5.14.2 Docker

From the Dataverse perspective, Docker is important for a few reasons:

- There is interest from the community in running Dataverse on OpenShift and some initial work has been done to get Dataverse running on Minishift in Docker containers. Minishift makes use of Docker images on Docker Hub. To build new Docker images and push them to Docker Hub, you'll need to install Docker. The main issue to follow is <https://github.com/IQSS/dataverse/issues/4040>.
- Docker may aid in testing efforts if we can easily spin up Docker images based on code in pull requests and run the full integration suite against those images. See the *Testing* section for more information on integration tests.

### Installing Docker

On Linux, you can probably get Docker from your package manager.

On Mac, download the `.dmg` from <https://www.docker.com> and install it. As of this writing it is known as Docker Community Edition for Mac.

On Windows, we have heard reports of success using Docker on a Linux VM running in VirtualBox or similar. There's something called "Docker Community Edition for Windows" but we haven't tried it. See also the [Windows Development](#) section.

As explained above, we use Docker images in two different contexts:

- Testing using an "all in one" Docker image (ephemeral, unpublished)
- Future production use on Minishift/OpenShift/Kubernetes (published to Docker Hub)

### All In One Docker Images for Testing

The "all in one" Docker files are in `conf/docker-ai-o` and you should follow the readme in that directory for more information on how to use them.

### Future production use on Minishift/OpenShift/Kubernetes

FIXME: rewrite this section to talk about only pushing stable images to Docker Hub.

When working with Docker in the context of Minishift, follow the instructions above and make sure you get the Dataverse Docker images running in Minishift before you start messing with them.

As of this writing, the Dataverse Docker images we publish under <https://hub.docker.com/u/iqss/> are highly experimental. They were originally tagged with branch names like `kick-the-tires` and as of this writing the latest tag should be considered highly experimental and not for production use. See <https://github.com/IQSS/dataverse/issues/4040> for the latest status and please reach out if you'd like to help!

Change to the docker directory:

```
cd conf/docker
```

Edit one of the files:

```
vim dataverse-glassfish/Dockerfile
```

At this point you want to build the image and run it. We are assuming you want to run it in your Minishift environment. We will be building your image and pushing it to Docker Hub.

Log in to Docker Hub with an account that has access to push to the `iqss` organization:

```
docker login
```

(If you don't have access to push to the `iqss` organization, you can push elsewhere and adjust your `openshift.json` file accordingly.)

Build and push the images to Docker Hub:

```
./build.sh
```

Note that you will see output such as `digest: sha256:213b6380e6ee92607db5d02c9e88d7591d81f4b6d713224d47` that should later be reflected in Minishift to indicate that you are using the latest image you just pushed to Docker Hub.

You can get a list of all repos under the `iqss` organization with this:



```
curl https://hub.docker.com/v2/repositories/iqss/
```

To see a specific repo:

```
curl https://hub.docker.com/v2/repositories/iqss/dataverse-glassfish/
```

## Known Issues with Dataverse Images on Docker Hub

Again, Dataverse Docker images on Docker Hub are highly experimental at this point. As of this writing, their purpose is primarily for kicking the tires on Dataverse. Here are some known issues:

- The Dataverse installer is run in the entrypoint script every time you run the image. Ideally, Dataverse would be installed in the Dockerfile instead. Dataverse is being installed in the entrypoint script because it needs PostgreSQL to be up already so that database tables can be created when the war file is deployed.
- The storage should be abstracted. Storage of data files and PostgreSQL data. Probably Solr data.
- Better tuning of memory by examining `/sys/fs/cgroup/memory/memory.limit_in_bytes` and incorporating this into the Dataverse installation script.
- Only a single Glassfish server can be used. See “Dedicated timer server in a Dataverse server cluster” in the *Dataverse Application Timers* section of the Installation Guide.
- Only a single PostgreSQL server can be used.
- Only a single Solr server can be used.

Previous: *Deployment* | Next: *Making Releases*

## 5.15 Making Releases

### Contents:

- *Create the release GitHub issue and branch*
  - *1. Bump Version Numbers*
  - *2. Check in the Changes Above...*
- *Merge “develop” into “master”*
- *Write Release Notes*
- *Make Artifacts Available for Download*
- *Publish Release*

Use the number of the milestone with a “v” in front for the release tag. For example: `v4.6.2`.

### 5.15.1 Create the release GitHub issue and branch

Use the GitHub issue number and the release tag for the name of the branch. For example: `4734-update-v-4.8.6-to-4.9`

**Note:** the changes below must be the very last commits merged into the develop branch before it is merged into master and tagged for the release!

Make the following changes in the release branch:

## 1. Bump Version Numbers

Increment the version number to the milestone (e.g. 4.6.2) in the following two files:

- pom.xml
- doc/sphinx-guides/source/conf.py (two places)

Add the version being released to the lists in the following two files:

- doc/sphinx-guides/source/versions.rst
- scripts/database/releases.txt

Here's an example commit where three of the four files above were updated at once: <https://github.com/IQSS/dataverse/commit/99e23f96ec362ac2f524cb5cd80ca375fa13f196>

## 2. Check in the Changes Above...

... into the release branch, make a pull request and merge the release branch into develop.

### 5.15.2 Merge “develop” into “master”

The “develop” branch should be merged into “master” before tagging. See also the branching strategy described in the *Version Control* section.

### 5.15.3 Write Release Notes

Developers should express the need for an addition to release notes by creating a file in `/doc/release-notes` containing the name of the issue they're working on. The name of the branch could be used for the filename with “.md” appended (release notes are written in Markdown) such as `5053-apis-custom-homepage.md`.

At or near release time:

- Create an issue in Github to track the work of creating release notes for the upcoming release
- Create a branch, add a .md file for the release (ex. 4.16 Release Notes) in `/doc/release-notes` and write the release notes, making sure to pull content from the issue-specific release notes mentioned above
- Delete the previously-created, issue-specific release notes as the content is added to the main release notes file
- Take the release notes .md through the regular Code Review and QA process
- Create a draft release at <https://github.com/IQSS/dataverse/releases/new>
- The “tag version” and “title” should be the number of the milestone with a “v” in front (i.e. v4.16).
- Copy in the content from the .md file
- For the description, follow post-4.16 examples at <https://github.com/IQSS/dataverse/releases>

### 5.15.4 Make Artifacts Available for Download

Upload the following artifacts to the draft release you created:

- war file (`mvn package` from Jenkins)
- installer (`cd scripts/installer && make`)
- other files as needed, such as updated Solr schema and config files

## 5.15.5 Publish Release

Click the “Publish release” button.

Previous: *Docker, Kubernetes, and OpenShift* | Next: *Tools*

## 5.16 Tools

These are handy tools for your *Development Environment*.

### Contents:

- *Netbeans Connector Chrome Extension*
- *pgAdmin*
- *Maven*
- *Vagrant*
- *PlantUML*
- *Eclipse Memory Analyzer Tool (MAT)*
- *PageKite*
- *MSV*
- *FontCustom*
- *SonarQube*
- *Infer*
- *lsof*
- *jmap and jstat*

### 5.16.1 Netbeans Connector Chrome Extension

The [Netbeans Connector](#) extension for Chrome allows you to see changes you’ve made to HTML pages the moment you save the file without having to refresh your browser. See also <http://wiki.netbeans.org/ChromeExtensionInstallation>

Unfortunately, while the Netbeans Connector Chrome Extension used to “just work”, these days a workaround described at <https://www.youtube.com/watch?v=J6lOQS2rWK0&t=130> seems to be necessary. For now, under “Run” (under project properties), choose “Chrome” as the browser rather than “Chrome with NetBeans Connector”. After you run the project, click the Netbeans logo in Chrome and then “Debug in NetBeans”. For more information, please see the “workaround for Netbeans Connector Chrome Extension” post at <https://groups.google.com/d/msg/dataverse-dev/agJZiID110Q/cMBkt5KDBQAJ>

### 5.16.2 pgAdmin

You probably installed pgAdmin when following the steps in the *Development Environment* section but if not, you can download it from <https://www.pgadmin.org>

### 5.16.3 Maven

With Maven installed you can run `mvn package` and `mvn test` from the command line. It can be downloaded from <https://maven.apache.org>

### 5.16.4 Vagrant

Vagrant allows you to spin up a virtual machine running Dataverse on your development workstation. You'll need to install Vagrant from <https://www.vagrantup.com> and VirtualBox from <https://www.virtualbox.org>.

We assume you have already cloned the repo from <https://github.com/IQSS/dataverse> as explained in the *Development Environment* section.

From the root of the git repo (where the `Vagrantfile` is), run `vagrant up` and eventually you should be able to reach an installation of Dataverse at <http://localhost:8888> (the `forwarded_port` indicated in the `Vagrantfile`).

Please note that running `vagrant up` for the first time should run the `downloads/download.sh` script for you to download required software such as Glassfish and Solr and any patches. However, these dependencies change over time so it's a place to look if `vagrant up` was working but later fails.

On Windows if you see an error like `/usr/bin/perl^M: bad interpreter` you might need to run `dos2unix` on the installation scripts.

### 5.16.5 PlantUML

PlantUML is used to create diagrams in the guides and other places. Download it from <http://plantuml.com> and check out an example script at <https://github.com/IQSS/dataverse/blob/v4.6.1/doc/Architecture/components.sh>. Note that for this script to work, you'll need the `dot` program, which can be installed on Mac with `brew install graphviz`.

### 5.16.6 Eclipse Memory Analyzer Tool (MAT)

The Memory Analyzer Tool (MAT) from Eclipse can help you analyze heap dumps, showing you “leak suspects” such as seen at <https://github.com/payara/Payara/issues/350#issuecomment-115262625>

It can be downloaded from <http://www.eclipse.org/mat>

If the heap dump provided to you was created with `gcore` (such as with `gcore -o /tmp/gf.core $glassfish_pid`) rather than `jmap`, you will need to convert the file before you can open it in MAT. Using `gf.core.13849` as example of the original 33 GB file, here is how you could convert it into a 26 GB `gf.core.13849.hprof` file. Please note that this operation took almost 90 minutes:

```
/usr/java7/bin/jmap -dump:format=b,file=gf.core.13849.hprof /usr/java7/bin/  
java gf.core.13849
```

A file of this size may not “just work” in MAT. When you attempt to open it you may see something like “An internal error occurred during: “Parsing heap dump from ‘/tmp/heapdumps/gf.core.13849.hprof’”. Java heap space”. If so, you will need to increase the memory allocated to MAT. On Mac OS X, this can be done by editing `MemoryAnalyzer.app/Contents/MacOS/MemoryAnalyzer.ini` and increasing the value “-Xmx1024m” until it's high enough to open the file. See also [http://wiki.eclipse.org/index.php/MemoryAnalyzer/FAQ#Out\\_of\\_Memory\\_Error\\_while\\_Running\\_the\\_Memory\\_Analyzer](http://wiki.eclipse.org/index.php/MemoryAnalyzer/FAQ#Out_of_Memory_Error_while_Running_the_Memory_Analyzer)

### 5.16.7 PageKite

PageKite is a fantastic service that can be used to share your local development environment over the Internet on a public IP address.

With PageKite running on your laptop, the world can access a URL such as <http://pdurbin.pagekite.me> to see what you see at <http://localhost:8080>

Sign up at <https://pagekite.net> and follow the installation instructions or simply download <https://pagekite.net/pk/pagekite.py>

The first time you run `./pagekite.py` a file at `~/pagekite.rc` will be created. You can edit this file to configure PageKite to serve up port 8080 (the default GlassFish HTTP port) or the port of your choosing.

According to <https://pagekite.net/support/free-for-foss/> PageKite (very generously!) offers free accounts to developers writing software the meets <http://opensource.org/docs/definition.php> such as Dataverse.

### 5.16.8 MSV

MSV (Multi Schema Validator) can be used from the command line to validate an XML document against a schema. Download the latest version from <https://java.net/downloads/msv/releases/> (msv.20090415.zip as of this writing), extract it, and run it like this:

```
$ java -jar /tmp/msv-20090415/msv.jar Version2-0.xsd ddi.xml
start parsing a grammar.
validating ddi.xml
the document is valid.
```

### 5.16.9 FontCustom

The custom file type icons were created with the help of *FontCustom* <<https://github.com/FontCustom/fontcustom>>. Their README provides installation instructions as well as directions for producing your own vector-based icon font.

Here is a vector-based SVG file to start with as a template: `icon-template.svg`

### 5.16.10 SonarQube

SonarQube is a static analysis tool that can be used to identify possible problems in the codebase, or with new code. It may report false positives or false negatives, but can help identify potential problems before they are reported in production or to identify potential causes of problems reported in production.

Download SonarQube from <https://www.sonarqube.org> and start look in the `bin` directory for a `sonar.sh` script for your architecture. Once the tool is running on <http://localhost:9000> you can use it as the URL in this example script to run sonar:

```
#!/bin/sh

mvn sonar:sonar \
-Dsonar.host.url=${your_sonar_url} \
-Dsonar.login=${your_sonar_token_for_project} \
-Dsonar.test.exclusions='src/test/**,src/main/webapp/resources/**' \
-Dsonar.issuesReport.html.enable=true \
-Dsonar.issuesReport.html.location='sonar-issues-report.html' \
-Dsonar.jacoco.reportPath=target/jacoco.exec
```

Once the analysis is complete, you should be able to access <http://localhost:9000/dashboard?id=edu.harvard.iq%3Adataverse> to see the report. To learn about resource leaks, for example, click on “Bugs”, the “Tag”, then “leak” or “Rule”, then “Resources should be closed”.

### 5.16.11 Infer

Infer is another static analysis tool that can be downloaded from <https://github.com/facebook/infer>

Example command to run infer:

```
$ infer -- mvn package
```

Look for “RESOURCE\_LEAK”, for example.

### 5.16.12 lsof

If file descriptors are not closed, eventually the open but unused resources can cause problems with system (glassfish in particular) stability. Static analysis and heap dumps are not always sufficient to identify the sources of these problems. For a quick sanity check, it can be helpful to check that the number of file descriptors does not increase after a request has finished processing.

For example...

```
$ lsof | grep M6EI0N | wc -l
0
$ curl -X GET "http://localhost:8083/dataset.xhtml?persistentId=doi:10.5072/FK2/
↪M6EI0N" > /dev/null
$ lsof | grep M6EI0N | wc -l
500
```

would be consistent with a file descriptor leak on the dataset page.

### 5.16.13 jmap and jstat

`jmap` and `jstat` are parts of the standard JDK distribution. `jmap` allows you to look at the contents of the java heap. It can be used to create a heap dump, that you can then feed to another tool, such as `Memory Analyzer Tool` (see above). It can also be used as a useful tool of its own, for example, to list all the classes currently instantiated in memory:

```
$ jmap -histo <glassfish process id>
```

will output a list of all classes, sorted by the number of instances of each individual class, with the size in bytes. This can be very useful when looking for memory leaks in the application. Another useful tool is `jstat`, that can be used in combination with `jmap` to monitor the effectiveness of garbage collection in reclaiming allocated memory.

In the example script below we stress running Dataverse application with GET requests to a specific dataverse page, use `jmap` to see how many `Dataverse`, `Dataset` and `DataFile` class object get allocated, then run `jstat` to see how the numbers are affected by both “Young Generation” and “Full” garbage collection runs (YGC and FGC respectively):

(This script is provided **as an example only!** You will have to experiment and expand it to suit any specific needs and any specific problem you may be trying to diagnose, and this is just to give an idea of how to go about it)

```
#!/bin/sh

# the script takes the numeric id of the glassfish process as the command line
↪argument:
```

```

id=$1

while :
do
  # Access the dataverse xxx 10 times in a row:
  for ((i = 0; i < 10; i++))
  do
    # hide the output, standard and stderr:
    curl http://localhost:8080/dataverse/xxx 2>/dev/null > /dev/null
  done

  sleep 1

  # run jmap and save the output in a temp file:

  jmap -histo $id > /tmp/jmap.histo.out

  # grep the output for Dataverse, Dataset and DataFile classes:
  grep '\.Dataverse$' /tmp/jmap.histo.out
  grep '\.Dataset$' /tmp/jmap.histo.out
  grep '\.DataFile$' /tmp/jmap.histo.out
  # (or grep for whatever else you may be interested in)

  # print the last line of the jmap output (the totals):
  tail -1 /tmp/jmap.histo.out

  # run jstat to check on GC:
  jstat -gcutil ${id} 1000 1 2>/dev/null

  # add a time stamp and a new line:

  date
  echo

done

```

The script above will run until you stop it, and will output something like:

```

439:          141          28200 edu.harvard.iq.dataverse.Dataverse
472:          160          24320 edu.harvard.iq.dataverse.Dataset
674:           60           9600 edu.harvard.iq.dataverse.DataFile
S0      S1      E      O      P      YGC      YGCT      FGC      FGCT      GCT
0.00 100.00 35.32 20.15  ?      7      2.145      0      0.000      2.145
Total      108808814      5909776392
Wed Aug 14 23:13:42 EDT 2019

385:          181          36200 edu.harvard.iq.dataverse.Dataverse
338:          320          48640 edu.harvard.iq.dataverse.Dataset
524:          120          19200 edu.harvard.iq.dataverse.DataFile
S0      S1      E      O      P      YGC      YGCT      FGC      FGCT      GCT
0.00 100.00 31.69 45.11  ?      9      3.693      0      0.000      3.693
Total      167998691      9080163904
Wed Aug 14 23:14:59 EDT 2019

367:          201          40200 edu.harvard.iq.dataverse.Dataverse
272:          480          72960 edu.harvard.iq.dataverse.Dataset
442:          180          28800 edu.harvard.iq.dataverse.DataFile
S0      S1      E      O      P      YGC      YGCT      FGC      FGCT      GCT

```

```
0.00 100.00 28.05 69.94 ? 11 5.001 0 0.000 5.001
Total 226826706 12230221352
Wed Aug 14 23:16:16 EDT 2019

... etc.
```

How to analyze the output, what to look for:

First, look at the numbers in the jmap output. In the example above, you can immediately see, after the first three iterations, that every 10 dataverse page loads results in the increase of the number of Dataset classes by 160. I.e., each page load leaves 16 of these on the heap. We can also see that each of the 10 page load cycles increased the heap by roughly 3GB; that each cycle resulted in a couple of YG (young generation) garbage collections, and in the old generation allocation being almost 70% full. These numbers in the example are clearly quite high and are an indication of some problematic memory allocation by the dataverse page - if this is the result of something you have added to the page, you probably would want to investigate and fix it. However, overly generous memory use **is not the same as a leak** necessarily. What you want to see now is how much of this allocation can be reclaimed by "Full GC". If all of it gets freed by FGC, it is not the end of the world (even though you do not want your system to spend too much time running FGC; it costs CPU cycles, and actually freezes the application while it's in progress!). It is however a **really** serious problem, if you determine that a growing portion of the old. gen. memory ("O" in the jmap output) is not getting freed, even by FGC. This *is* a real leak now, i.e. something is leaving behind some objects that are still referenced and thus off limits to garbage collector. So look for the lines where the FGC counter is incremented. For example, the first FGC in the example output above:

```
271:          487          97400 edu.harvard.iq.dataverse.Dataverse
216:         3920         150784 edu.harvard.iq.dataverse.Dataset
337:          372         59520 edu.harvard.iq.dataverse.DataFile
Total 277937182 15052367360
S0  S1  E  O  P  YGC  YGCT  FGC  FGCT  GCT
0.00 100.00 77.66 88.15 ? 17 8.734 0 0.000 8.734
Wed Aug 14 23:20:05 EDT 2019

265:          551         110200 edu.harvard.iq.dataverse.Dataverse
202:         4080         182400 edu.harvard.iq.dataverse.Dataset
310:          450         72000 edu.harvard.iq.dataverse.DataFile
Total 142023031 8274454456
S0  S1  E  O  P  YGC  YGCT  FGC  FGCT  GCT
0.00 100.00 71.95 20.12 ? 22 25.034 1 4.455 29.489
Wed Aug 14 23:21:40 EDT 2019
```

We can see that the first FGC resulted in reducing the "O" by almost 7GB, from 15GB down to 8GB (from 88% to 20% full). The number of Dataset classes has not budged at all - it has grown by the same 160 objects as before (very suspicious!). To complicate matters, FGC does not **guarantee** to free everything that can be freed - it will balance how much the system needs memory vs. how much it is willing to spend in terms of CPU cycles performing GC (remember, the application freezes while FGC is running!). So you should not assume that the "20% full" number above means that you have 20% of your stack already wasted and unrecoverable. Instead, look for the next **minium** value of "O"; then for the next, etc. Now compare these consecutive miniums. With the above test (this is an output of a real experiment, a particularly memory-hungry feature added to the dataverse page), the minimums sequence (of old. gen. usage, in %) was looking as follows:

```
2.19
2.53
3.00
3.13
3.95
4.03
4.21
```



```
4.40
4.64
5.06
5.17
etc. ...
```

It is clearly growing - so now we can conclude that indeed something there is using memory in a way that's not recoverable, and this is a clear problem.

Previous: *Making Releases* | Next: *Universal Numerical Fingerprint (UNF)*

## 5.17 Universal Numerical Fingerprint (UNF)

$$\begin{pmatrix} 1 & 4 & 4 & 21 & \dots & 121 \\ 1 & 2 & 2 & 91 & \dots & 212 \\ 1 & 9 & 2 & 72 & \dots & 104 \\ 0 & 2 & 2 & 2 & \dots & 321 \\ 1 & 6 & 2 & 12 & \dots & 204 \\ 1 & 9 & 4 & 52 & \dots & 311 \\ 0 & 3 & 2 & 23 & \dots & 92 \\ 0 & 2 & 5 & 91 & \dots & 212 \\ 0 & 5 & 8 & 91 & \dots & 91 \\ 1 & 9 & 1 & 72 & \dots & 104 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 2 & 91 & \dots & 212 \end{pmatrix}$$


- Apply a **cryptographic algorithm**
- Solely based on semantic contents of the digital object:
  - **data changes** result in **new UNF**
  - **format or location changes** retain **original UNF**
- Final alphanumeric string:
  - **uniquely** summarizes the contents,
  - but does **not convey its information**



**ZNQRI14053UZq389x0Bffg?==**

Fig. 5.1: Fig.1 UNF: used to uniquely identify and verify data.

Universal Numerical Fingerprint (UNF) is a unique signature of the **semantic content** of a digital object. It is **not** simply a checksum of a binary data file. Instead, the UNF algorithm approximates and normalizes the data stored within. A cryptographic hash of that normalized (or canonicalized) representation is then computed. The signature is thus independent of the storage format. E.g., the same data object stored in, say, SPSS and Stata, will have the same UNF.

Early versions of Dataverse were using the first released implementation of the UNF algorithm (v.3, implemented in R). Starting with Dataverse 2.0 and throughout the 3.\* lifecycle, UNF v.5 (implemented in Java) was used. Dataverse 4.0 uses the latest release, UNF v.6. Two parallel implementation, in R and Java, will be available, for cross-validation.

Learn more: Micah Altman and Gary King. 2007. "A Proposed Standard for the Scholarly Citation of Quantitative Data." D-Lib Magazine, 13. Publisher's Version Copy at <http://j.mp/2ovSzoT>

**Contents:****5.17.1 UNF Version 3**

Version 3 of the UNF algorithm was used by the Dataverse Network software prior to version 2.0, and was implemented in R code. This algorithm was used on digital objects containing vectors of numbers, vectors of character strings, data sets comprising such vectors, and studies comprising one or more such data sets.

The UNF V3 algorithm applied to the content of a data set or study is as follows:

1. Round each element in a numeric vector to  $k$  significant digits using the IEEE 754 round towards zero rounding mode. The default value of  $k$  is seven, the maximum expressible in single-precision floating point calculations. UNF calculation for vectors of character strings is identical, except that you truncate to  $k$  characters and the default value of  $k$  is 128.
2. Convert each vector element to a character string in exponential notation, omitting noninformational zeros. If an element is missing, represent it as a string of three null characters. If an element is an IEEE 754, nonfinite, floating-point special value, represent it as the signed, lowercase, IEEE minimal printable equivalent (that is, `+inf`, `-inf`, or `+nan`).

Each character string comprises the following:

- A sign character.
- A single leading digit.
- A decimal point.
- Up to  $k-1$  digits following the decimal, consisting of the remaining  $k-1$  digits of the number, omitting trailing zeros.
- A lowercase letter “e.”
- A sign character.
- The digits of the exponent, omitting trailing zeros.

For example, the number pi at five digits is represented as `-3.1415e+`, and the number 300 is represented as the string `+3.e+2`.

1. Terminate character strings representing nonmissing values with a POSIX end-of-line character.
2. Encode each character string with [Unicode bit encoding](#). Versions 3 through 4 use UTF-32BE; Version 4.1 uses UTF-8.
3. Combine the vector of character strings into a single sequence, with each character string separated by a POSIX end-of-line character and a null byte.
4. Compute a hash on the resulting sequence using the standard MD5 hashing algorithm for Version 3 and using [SHA256](#) for Version 4. The resulting hash is [base64](#) encoded to support readability.
5. Calculate the UNF for each lower-level data object, using a consistent UNF version and level of precision across the individual UNFs being combined.
6. Sort the base64 representation of UNFs in POSIX locale sort order.
7. Apply the UNF algorithm to the resulting vector of character strings using  $k$  at least as large as the length of the underlying character string.
8. Combine UNFs from multiple variables to form a single UNF for an entire data frame, and then combine UNFs for a set of data frames to form a single UNF that represents an entire research study.

Learn more: Software for computing UNFs is available in an R Module, which includes a Windows standalone tool and code for Stata and SAS languages. Also see the following for more details: Micah Altman and Gary King. 2007. “A Proposed Standard for the Scholarly Citation of Quantitative Data,” D-Lib Magazine, Vol. 13, No. 3/4 (March). (Abstract: [HTML](#) | Article: [PDF](#))

## 5.17.2 UNF Version 5

### Important Update:

UNF Version 5 has been in use by the Dataverse project since 2009. It was built into every version of the DVN, starting with 2.0 and up to 3.6.2. However, some problems were recently found in that implementation. Namely, in certain cases data normalization is not implemented fully to the spec. UNF signatures it generates are still reasonably strong statistically; however, this means that at least some of our signatures are not independently verifiable. I.e., if somebody fully implements their own version of UNF calculator, for certain datasets it would calculate signatures different from those generated by the DVN. Unless of course they implement it with the exact same bugs as ours.

To address this, the Project is about to release UNF Version 6. The release date is still being discussed. It may coincide with the release of Dataverse 4.0. Alternatively, the production version of DVN 3.6.3 may get upgraded to use UNF v6 prior to that. This will be announced shortly. In the process, we are solving another problem with UNF v5 - this time we’ve made an effort to offer very implementer-friendly documentation that describes the algorithm fully and unambiguously. So if you are interested in implementing your own version of a UNF calculator, (something we would like to encourage!) please proceed directly to the Version 6 documentation.

Going forward, we are going to offer a preserved version of the Version 5 library and, possibly, an online UNF v5 calculator, for the purposes of validating vectors and data sets for which published Version 5 UNFs exist.

## 5.17.3 UNF Version 6

*(this document is a draft!)*

The document is primarily intended for those who are interested in implementing their own UNF Version 6 calculator. We would like to encourage multiple parallel implementations, since that would be a great (the only, really) way to cross-validate UNF signatures calculated for specific sets of data.

### Algorithm Description

UNF v5, on which v6 is based, was originally described in Dr. Micah Altman’s paper “A Fingerprint Method for Verification of Scientific Data”, Springer Verlag, 2008. The reader is encouraged to consult it for the explanation of the theory behind UNF. However, various changes and clarifications concerning the specifics of normalization have been made to the algorithm since the publication. These crucial details were only documented in the author’s unpublished edits of the article and in private correspondence. With this document, a serious effort has been made to produce a complete step-by-step description of the entire process. It should be fully sufficient for the purposes of implementing the algorithm.

#### Contents:

- *I. UNF of a Data Vector*
- *II. Combining multiple UNFs to create UNFs of higher-level objects.*
- *Footnotes:*

## I. UNF of a Data Vector

For each individual vector in a data frame, calculate its UNF signature as follows:

### Ia. Normalize each vector element as follows:

**1. For a vector of numeric elements:** Round each vector element to  $N$  significant digits using the IEEE 754 “round towards nearest, ties to even” rounding mode. The default value of  $N$  is 7.

(See an Important *Note* on the use of *default and optional* values and methods!)

Convert each vector element into a character string in exponential notation, as follows:

A sign character.

A single leading non-zero digit.

A decimal point.

Up to  $N-1$  remaining digits following the decimal, no trailing zeros.

A lowercase letter “e”.

A sign character.

The digits of the exponent, omitting trailing zeros.

*Special cases:*

Zero representation (an exception to the “leading non-zero digit” rule, above):

+0.e+ for positive zero.

-0.e+ for negative zero.

(see the *Note* below on *negative zero*)

Infinity and NaN (“Not a Number”) values:

If an element is an IEEE 754, non-finite, special floating-point value, represent it as the signed, lowercase, IEEE minimal printable equivalent, that is, `+inf`, `-inf`, or `+nan`. No attempt is made to differentiate between various types of NaNs allowed under IEEE 754.

*Examples:*

The number 1 is normalized as `+1.e+`

The number  $\pi$  at 5 digits is normalized as `+3.1415e+`

The number  $-300$  is normalized as `-3.e+2`

The number  $0.00073$  is normalized as `+7.3e-4`

Positive infinity is normalized as `+inf`

The number  $1.23456789$ , normalized with the default rounding to 7 digits of precision, is normalized as `+1.234568e+`

An “official” list of pre-calculated sample UNFs is supplied with the source of the Java implementation of UNF v6; see the *Note* at the end of the document.

### 2. For a vector of character strings:

Encode each character string with Unicode bit encoding. In UNF Version 6 UTF-8 is used. Truncate each string to X characters; the default value of X is 128. No further normalization is performed.

### 3. Vectors of Boolean values

Should be treated as numeric vectors of 0 s and 1 s.

### 4. Bit fields.

Normalize bit fields by converting to big-endian form, truncating all leading empty bits, aligning to a byte boundary by padding with leading zero bits, and base64 encoding to form a character string representation.

### 5. Normalize dates, times and intervals as follows:

#### 5a. Dates.

Convert calendar dates to a character string of the form YYYY-MM-DD, zero padded. Partial dates in the form YYYY or YYYY-MM are permitted

#### 5b. Time.

Time representation is based on an ISO 8601 format hh:mm:ss.ffffff. hh, mm and ss are 2 digit, zero-padded numbers. fffffff represents fractions of a second, it must contain no trailing (non-significant) zeroes, and must be omitted altogether the value is zero. No other fractional representations, such as fractional minutes or hours, are permitted. If the time zone of the observation is known, convert the time value to the UTC time zone and append a "Z" to the time representation. (In other words, no time zones other than UTC are allowed in the final normalized representation).

(see the *Note* at the end of this document for a discussion on *potential issues when calculating UNFs of time values*).

#### 5c. Combined Date and Time values.

Format elements that comprise a combined date and time by concatenating the (full) date representation, a single letter "T", and the time representation. Partial date representations are **prohibited** in combined date and time values.

#### 5d. Intervals.

Represent intervals by using two date-time values, each formatted as defined previously, and separated by a slash ("/").

*Durations*, that were mentioned in the old UNF v5 document are NOT in fact implemented and have been dropped from the spec.

#### Examples:

2:29 pm on Jun 10, 2012 is normalized as "2012-06-10T14:29:00".

Fri Aug 22 12:51:05 EDT 2014 is normalized as "2014-08-22T16:51:05Z"  
(The UTC offset of Eastern Daylight Time is -4:00).

**6. Missing values** Missing values, of all of the above types, are encoded as 3 null bytes: \000\000\000.

**Ib. Calculate the UNF of the vector as follows:**

Terminate each character string representing a NON-MISSING value with a POSIX end-of-line character and a null byte (\000). Do not terminate missing value representations (3 null bytes \000\000\000). Concatenate all the individual character strings, and compute the SHA256 hash of the combined string. Truncate the resulting hash to 128 bits (128 being the default, with other values possible - see the note at the end of the document). Encode the resulting string in base64, for readability. Prepend the encoded hash string with the signature header UNF : 6 : (with 6 indicating the current version).

*Example:*

Vector (numeric): {1.23456789, <MISSING VALUE>, 0}

Normalized elements (N = 7,default): "+1.234568e+", "\000\000\000", "+0.e+"

Combined string: "+1.234568e+\n\000\000\000\000+0.e+\n\000"

SHA256 hash, truncated to the default 128 bits: Do5dfAo00Ft4FSj0JcByEw==

Printable UNF: UNF : 6 : Do5dfAo00Ft4FSj0JcByEw==

**II. Combining multiple UNFs to create UNFs of higher-level objects.****IIa. Combine the UNFs of multiple variables to form the UNF for an entire data frame as follows:**

*UNF of a data frame (datafile) with 1 variable:*

The UNF of the data frame is the same as the UNF of the variable.

*UNF of a data frame with the number of variables > 1:*

Sort the printable UTF8 representations of the individual UNFs in the POSIX locale sort order.

Apply the UNF algorithm to the resulting vector of character strings.

Do note the **sorting** part, above, it is important! In a vector of observations, the order is important; changing the order of observations changes the UNF. A data frame, however, is considered an unordered set of individual vectors. I.e., re-arranging the order in which data variable columns occur in an R or Stata file should not affect the UNF. Hence the UNFs of individual variables are sorted, before the combined UNF of the data frame is calculated.

**IIb. Similarly, combine the UNFs for a set of data frames to form a single UNF that represents an entire research study ("dataset").**

Again, the UNF of a study (dataset) with a single file = the UNF of the file; for more than one file, calculate the study UNF as described above.

Using a consistent UNF version and level of precision across an entire dataset is recommended when calculating the UNFs of individual data objects.

**Footnotes:**

**Note: On default and non-default parameter values:** Here and throughout the rest of this document, phrases like "The default value of N is 7" suggest that it is possible to use non-default values, such as a different number of digits of precision, in this case. This has been a source of some confusion in the past. UNF relies on data normalization to produce "data fingerprints" that are meaningful and descriptive. So how do you generate reproducible and verifiable signatures if any flexibility is allowed in the normalization algorithm? The answer, as specified in the original UNF paper: any non-default parameters used are embedded in the header portion of the UNF!

For example, to specify a non-default precision the parameter it is specified using the parameter N, formatted as follows:

Nnnn - where nnn is the number of precision digits, different from the default 7.

*Example:*

The UNF of a floating point (Double) vector with a single element of 1.23456789, calculated with the default 7 digits of precision, is UNF : 6 : vcKELUSS4s4k1snF4OTB9A==. If we want to calculate the signature with N = 9, the resulting printable UNF is UNF : 6 : N9 : IKw+14ywdwsJeDze8dp1JA==. With the parameter value embedded in the signature, it can be recalculated and verified unambiguously.

Other optional parameters supported:

**(multiple parameters are added comma-separated, in any order)**

X### - where ### is the number of bytes for truncation of character strings;  
128 is the default.

H### - where ### is the number of bits to which the SHA256 hash should be truncated.

Allowed values are {128 , 192 , 196 , 256} with 128 being the default.

R1 - **truncate** numeric values to N digits, **instead of rounding**, as previously described.

Dr. Micah Altman's classic UNF v5 paper mentions another optional parameter T###, for specifying rounding of date and time values (implemented as stripping the values of entire components - fractional seconds, seconds, minutes, hours... etc., progressively) - but it doesn't specify its syntax. It is left as an exercise for a curious reader to contact the author and work out the details, if so desired. (Not implemented in UNF Version 6 by the Dataverse Project).

Note: we do not recommend truncating character strings at fewer bytes than the default 128 (the X parameter). At the very least this number **must** be high enough so that the printable UNFs of individual variables or files are not truncated, when calculating combined UNFs of files or datasets, respectively.

It should also be noted that the Dataverse application never calculates UNFs with any non-default parameters. And we are not aware of anyone else actually doing so. If you are considering creating your own implementation of the UNF, it may be worth trying to create a simplified, defaults-only version first. Such an implementation would be sufficient to independently verify Dataverse-produced UNFs, among other things. **Note: Negative Zero**

IEEE 754 zero is signed. I.e., there are 2 zeros, positive and negative. As implemented in most programming languages, floating point types can have negative zero values. It is the responsibility of the implementer, to properly identify the sign of a floating point zero value. Which can be a bit tricky; for example, in Java programming language, when performing arithmetic comparison on values of the primitive type `double`, the following evaluates to `TRUE`:

```
0.0d == -0.0d
```

However, Java also provides a wrapper class `Double`, with comparison methods that recognize `-0.0` and `0.0` as different values, and `0.0` to be greater than `-0.0`. So all of the following expressions evaluate to `FALSE`:

```
new Double(0.0d).equals(new Double(-0.0d))
```

```
Double.compare(-0.0d, 0.0d) >= 0
```

```
new Double(-0.0d).compareTo(new Double(0.0d)) >= 0
```

**Note: UNFs of time values in real-life statistical packages**

The following is not by itself an implementation concern. But it is something you may need to consider when calculating UNFs of time values from real-world data.

The fact that the same time value with and without the time zone specified produces different UNFs presents an interesting issue when converting data between different formats. For example, in STATA none of the available time types support time zones. In R, on the other hand, ALL time values are stored with a time zone. While it is possible to create an R time value from a character representation with no time zone - for example:

```
timevar<-as.POSIXct("03/19/2013 18:20:00", format = "%m/%d/%Y %H:%M:%OS");
```

it still results in R assuming the time is in the **current** time zone, and storing the UTC equivalent of that time. In fact R always stores its time values in UTC; specific time zones can be defined, as attributes, in which case the values will be adjusted accordingly for display. Otherwise the display representation will be readjusted each time the vector is viewed, according to the time zone **current to the viewer**. Meaning that the human readable representation of the same stored time value will be different when viewed on systems in different time zones. With that in mind, it appears that the only way to calculate a meaningful UNF of a time value from an R data frame is to use the stored UTC time - resulting in the “Z” in the normalized string. And that further means that it is impossible to convert a data frame with time values from STATA to R, or the other way around, and have the same UNF preserved.

We do not consider this a problem with the algorithm. These differences between the two approaches to handling time values, in R and STATA, should in fact be considered as **significant**. Enough so to conclude that the format conversion actually changes the data **semantically**. Which, in turn, justifies a new UNF.

If for whatever reason it is important to produce an R version of a STATA file while preserving the UNF, it can still be done. One way to achieve that would be to convert the original time vector to a String vector in R, in the format identical to that used in the UNF normalization algorithm, e.g., “yy-mm-ddThh:mm:ss”. One would not be able to use this resulting R vector in any time-based calculations without extra type conversion. But the data frame would produce the same UNF. **More UNF Examples:**

An “official” [list of sample UNFs](#) of various data types is provided with the source of the UNF v6 Java implementation.

---

Previous: [Tools](#) | Next: [Shibboleth and OAuth](#)

## 5.18 Make Data Count

Support for Make Data Count is a feature of Dataverse that is described in the [Make Data Count](#) section of the Admin Guide. In order for developers to work on the feature, they must install Counter Processor, a Python 3 application, as described below. Counter Processor can be found at <https://github.com/CDLUC3/counter-processor>

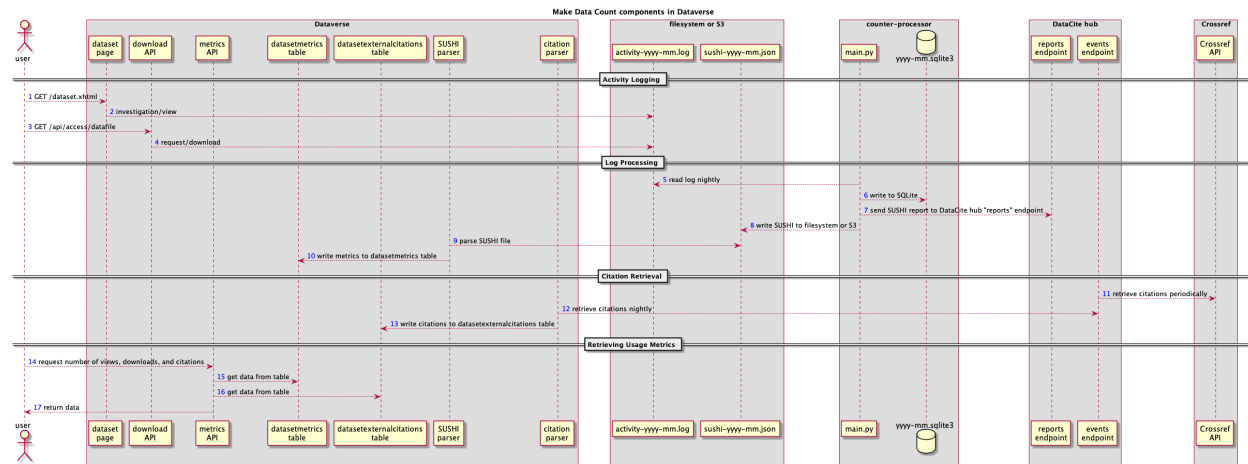
### Contents:

- [Architecture](#)
- [Dev Environment Setup for Make Data Count](#)
  - [Generate Fake Metrics Only](#)
  - [Full Setup](#)
- [Testing Make Data Count and Dataverse](#)
- [Resources](#)



## 5.18.1 Architecture

There are many components involved in Dataverse’s architecture for Make Data Count as shown in the diagram below.



## 5.18.2 Dev Environment Setup for Make Data Count

### Generate Fake Metrics Only

The quickest way to put populate tables with some data is to run the `MakeDataCountApiIT` integration test. For more on running integration tests see the [Testing](#) section. This will populate views and downloads.

To insert a citation you could insert a row like below, changing “72” in the example below with the dataset id for your dataset.

```
INSERT INTO datasetexternalcitations table (id, citedbyurl, dataset_id) VALUES (1, 'https://doi.org/10.1007/s12520-017-0578-2', 72);
```

### Full Setup

The recommended way to work on the Make Data Count feature is to spin up an EC2 instance that has both Dataverse and Counter Processor installed. Go to the [Deployment](#) page for details on how to spin up an EC2 instance and make sure that your Ansible file is configured to install Counter Processor before running the “create” script.

(Alternatively, you can try installing Counter Processor in Vagrant. `setup-counter-processor.sh` might help you get it installed.)

After you have spun to your EC2 instance, set `:MDCLogPath` so that Dataverse creates a log for Counter Processor to operate on. For more on this database setting, see the [Configuration](#) section of the Installation Guide.

Next you need to have Dataverse add some entries to the log that Counter Processor will operate on. To do this, click on some published datasets and download some files.

Next you should run Counter Processor to convert the log into a SUSHI report, which is in JSON format. Before running Counter Processor, you need to put a configuration file into place. As a starting point use `counter-processor-config.yaml` and edit the file, paying particular attention to the following settings:

- `log_name_pattern` You might want something like `/usr/local/glassfish4/glassfish/domains/domain1/logs/counter_(yyyy-mm-dd).log`
- `year_month` You should probably set this to the current month.

- `output_file` This needs to be a directory that the “glassfish” Unix user can read but that the “counter” user can write to. In dev, you can probably get away with “/tmp” as the directory.
- `platform` Out of the box from Counter Processor this is set to `Dash` but we’re not 100% sure if this should be “Dataverse” or a branch for a Dataverse installation like “LibreScholar”.
- `upload_to_hub` This should be “False” unless you are testing sending SUSHI reports to the DataCite hub.
- `simulate_date` You should probably set this to tomorrow.

Once you are done with your configuration, you can run Counter Processor like this:

```
sudo -i
su - counter
cd /usr/local/counter-processor-0.0.1
CONFIG_FILE=counter-processor-config.yaml python36 main.py
```

(Please note that the Counter Processor README says you can also pass in values like `START_DATE`, `END_DATE` etc. at the command line if you find this to be more convenient.)

After `main.py` has finished, you should see output that a JSON file has been placed in the directory you specified in `output_file`. Next, pass this JSON file to `curl` like this, substituting the DOI of the dataset you’re testing with:

```
curl -X POST 'http://localhost:8080/api/admin/makeDataCount/:persistentId/
addUsageMetricsFromSushiReport?reportOnDisk=/tmp/sushi_sample_logs.
json&persistentId=doi:10.5072/FK2/BL2IBM'
```

(Note that in production the `persistentId` of a dataset is not passed. Instead the PIDs for the datasets are pulled out of the JSON file.)

Assuming the `curl` command above worked, you should be able to retrieve the views and downloads via API like this, substituting the metric you’re interested in (`viewsTotal`, `viewsUnique`, `downloadsTotal`, `downloadsUnique`) and the DOI of the dataset you’re testing with:

```
curl -X POST 'http://localhost:8080/api/datasets/:persistentId/makeDataCount/
viewsTotal/2019-01&persistentId=doi:10.5072/FK2/BL2IBM'
```

If all this is working and you want to send data to the test instance of the Datacite hub, change `upload_to_hub` to “True” and contact [support@datacite.org](mailto:support@datacite.org) to get a JSON Web Token (JWT) to test with. Counter Processor should send the SUSHI reports for you but if you need to troubleshoot sending the reports manually, you can try the following `curl` command, substituting your JWT:

```
curl --header "Content-Type: application/json; Accept: application/json" -H
"Authorization: Bearer $JSON_WEB_TOKEN" -X POST https://api.test.datacite.
org/reports/ -d @sushi_report.json
```

For how to put citations into your dev database and how to get them out again, see “Configuring Dataverse for Make Data Count Citations” in the *Make Data Count* section of the Admin Guide.

### 5.18.3 Testing Make Data Count and Dataverse

A developer running Counter Processor alongside Dataverse for development or testing purposes will notice that once the raw Dataverse logs have been processed, there is no straightforward way to re-test those same logs.

The first thing to fix is to clear two files from Counter Processor `state` folder, `statefile.json` and `counter_db_[yyyy-mm].sqlite3`

Second, if you are also sending your SUSHI report to Make Data Count, you will notice that re-running Counter Processor will not update the information logged on their servers. This is due to us clearing the state of Counter

Processor, which in turn causes Counter Processor to send incorrectly structured requests to Make Data Count. The easiest way to resolve this issue is to DELETE the record Counter Processor has created on Make Data Count:

```
curl -H "Authorization: Bearer $JSON_WEB_TOKEN" -X DELETE https://
$MDC_SERVER/reports/$REPORT_ID
```

To get the REPORT\_ID, look at the logs generated in /usr/local/counter-processor-0.0.1/tmp/datacite\_response\_body.txt

To read more about the Make Data Count api, see <https://github.com/datacite/sashimi>

## 5.18.4 Resources

The original issue for adding Make Data Count support is <https://github.com/IQSS/dataverse/issues/4821>

## 5.19 Shibboleth and OAuth

### Contents:

- *Shibboleth and OAuth*

### 5.19.1 Shibboleth and OAuth

If you are working on anything related to users, please keep in mind that your changes will likely affect Shibboleth and OAuth users. For some background on user accounts in Dataverse, see “Auth Modes: Local vs. Remote vs. Both” in the *Configuration* section of the Installation Guide.

Rather than setting up Shibboleth on your laptop, developers are advised to simply add a value to their database to enable Shibboleth “dev mode” like this:

```
curl http://localhost:8080/api/admin/settings/:DebugShibAccountType -X PUT -d
RANDOM
```

For a list of possible values, please “find usages” on the settings key above and look at the enum.

Now when you go to <http://localhost:8080/shib.xhtml> you should be prompted to create a Shibboleth account.

OAuth is much more straightforward to get working on your laptop than Shibboleth. GitHub is a good identity provider to test with because you can easily request a Client ID and Client Secret that works against localhost. Follow the instructions in the *OAuth Login: ORCID, GitHub, Google* section of the installation Guide and use “<http://localhost:8080/oauth2/callback.xhtml>” as the callback URL.

In addition to setting up OAuth on your laptop for real per above, you can also use a dev/debug mode:

```
curl http://localhost:8080/api/admin/settings/:DebugOAuthAccountType -X PUT -d
RANDOM_EMAIL2
```

For a list of possible values, please “find usages” on the settings key above and look at the enum.

Now when you go to <http://localhost:8080/oauth2/firstLogin.xhtml> you should be prompted to create a Shibboleth account.

---

Previous: *Universal Numerical Fingerprint (UNF)* | Next: *Geospatial Data*

## 5.20 Geospatial Data

### Contents:

- *Geoconnect*
- *How Dataverse Ingests Shapefiles*
  - *Ingest*
  - *Example*
- *WorldMap JoinTargets + API Endpoint*
  - *How Geoconnect Uses Join Target Information*
  - *Retrieving Join Target Information from WorldMap API*
  - *Saving Join Target Information to Geoconnect Database*
  - *Setting Up WorldMap Test Data*

### 5.20.1 Geoconnect

Geoconnect works as a middle layer, allowing geospatial data files in Dataverse to be visualized with Harvard WorldMap. To set up a Geoconnect development environment, you can follow the steps outlined in the [local\\_setup.md](#) guide. You will need Python and a few other prerequisites.

As mentioned under “Architecture and Components” in the [Preparation](#) section of the Installation Guide, Geoconnect is an optional component of Dataverse, so this section is only necessary to follow it you are working on an issue related to this feature.

### 5.20.2 How Dataverse Ingests Shapefiles

A shapefile is a set of files, often uploaded/transferred in `.zip` format. This set may contain up to fifteen files. A minimum of three specific files (`.shp`, `.shx`, `.dbf`) are needed to be a valid shapefile and a fourth file (`.prj`) is required for WorldMap – or any type of meaningful visualization.

For ingest and connecting to WorldMap, four files are the minimum required:

- `.shp` - shape format; the feature geometry itself
- `.shx` - shape index format; a positional index of the feature geometry to allow seeking forwards and backwards quickly
- `.dbf` - attribute format; columnar attributes for each shape, in dBase IV format
- `.prj` - projection format; the coordinate system and projection information, a plain text file describing the projection using well-known text format

#### Ingest

When uploaded to Dataverse, the `.zip` is unpacked (same as all `.zip` files). Shapefile sets are recognized by the same base name and specific extensions. These individual files constitute a shapefile set. The first four are the minimum required (`.shp`, `.shx`, `.dbf`, `.prj`)

For example:

- bicycles.shp (required extension)
- bicycles.shx (required extension)
- bicycles.prj (required extension)
- bicycles.dbf (required extension)
- bicycles.sbx (NOT required extension)
- bicycles.sbn (NOT required extension)

Upon recognition of the four required files, Dataverse will group them as well as any other relevant files into a shapefile set. Files with these extensions will be included in the shapefile set:

- Required: .shp, .shx, .dbf, .prj
- Optional: .sbn, .sbx, .fbn, .fbx, .ain, .aih, .ixs, .mxs, .atx, .cpg, shp.xml

Then Dataverse creates a new .zip with mimetype as a shapefile. The shapefile set will persist as this new .zip.

## Example

### 1a. Original .zip contents:

A file named `bikes_and_subways.zip` is uploaded to the Dataverse. This .zip contains the following files.

- bicycles.shp (shapefile set #1)
- bicycles.shx (shapefile set #1)
- bicycles.prj (shapefile set #1)
- bicycles.dbf (shapefile set #1)
- bicycles.sbx (shapefile set #1)
- bicycles.sbn (shapefile set #1)
- bicycles.txt
- the\_bikes.md
- readme.txt
- subway\_line.shp (shapefile set #2)
- subway\_line.shx (shapefile set #2)
- subway\_line.prj (shapefile set #2)
- subway\_line.dbf (shapefile set #2)

### 1b. Dataverse unzips and re-zips files:

Upon ingest, Dataverse unpacks the file `bikes_and_subways.zip`. Upon recognizing the shapefile sets, it groups those files together into new .zip files:

- files making up the “bicycles” shapefile become a new .zip
- files making up the “subway\_line” shapefile become a new .zip
- remaining files will stay as they are

To ensure that a shapefile set remains intact, individual files such as `bicycles.sbn` are kept in the set – even though they are not used for mapping.

### 1c. Dataverse final file listing:

- bicycles.zip (contains shapefile set #1: bicycles.shp, bicycles.shx, bicycles.prj, bicycles.dbf, bicycles.sbx, bicycles.sbn)
- bicycles.txt (separate, not part of a shapefile set)
- the\_bikes.md (separate, not part of a shapefile set)
- readme.txt (separate, not part of a shapefile set)
- subway\_line.zip (contains shapefile set #2: subway\_line.shp, subway\_line.shx, subway\_line.prj, subway\_line.dbf)

For two “final” shapefile sets, bicycles.zip and subway\_line.zip, a new mimetype is used:

- Mimetype: application/zipped-shapefile
- Mimetype Label: “Shapefile as ZIP Archive”

### 5.20.3 WorldMap JoinTargets + API Endpoint

WorldMap supplies target layers – or JoinTargets – that a tabular file may be mapped against. A JSON description of these CGA-curated JoinTargets may be retrieved via API at <http://worldmap.harvard.edu/datatables/api/jointargets/>. Please note: login is required. You may use any WorldMap account credentials via HTTP Basic Auth.

Example of JoinTarget information returned via the API:

```
{
  "data": [
    {
      "layer": "geonode:census_tracts_2010_boston_6f6",
      "name": "Census Tracts, Boston (GEOID10: State+County+Tract)",
      "geocode_type_slug": "us-census-tract",
      "geocode_type": "US Census Tract",
      "attribute": {
        "attribute": "CT_ID_10",
        "type": "xsd:string"
      },
      "abstract": "As of the 2010 census, Boston, MA contains 7,288 city blocks_
↪ [truncated for example]",
      "title": "Census Tracts 2010, Boston (BARI)",
      "expected_format": {
        "expected_zero_padded_length": -1,
        "is_zero_padded": false,
        "description": "Concatenation of state, county and tract for 2010 Census_
↪ Tracts. Reference: https://www.census.gov/geo/maps-data/data/tract_rel_layout.
↪ html\r\n\r\nNote: Across the US, this can be a zero-padded \"string\" but the_
↪ original Boston layer has this column as \"numeric\" ",
        "name": "2010 Census Boston GEOID10 (State+County+Tract)"
      },
      "year": 2010,
      "id": 28
    },
    {
      "layer": "geonode:addresses_2014_boston_1wr",
      "name": "Addresses, Boston",
      "geocode_type_slug": "boston-administrative-geography",
      "geocode_type": "Boston, Administrative Geography",
      "attribute": {
        "attribute": "LocationID",

```

```

        "type": "xsd:int"
      },
      "abstract": "Unique addresses present in the parcels data set, which itself_
↪ is derived from [truncated for example]",
      "title": "Addresses 2015, Boston (BARI)",
      "expected_format": {
        "expected_zero_padded_length": -1,
        "is_zero_padded": false,
        "description": "Boston, Administrative Geography, Boston Address Location_
↪ ID. Example: 1, 2, 3...nearly 120000",
        "name": "Boston Address Location ID (integer)"
      },
      "year": 2015,
      "id": 18
    },
    {
      "layer": "geonode:bra_neighborhood_statistical_areas_2012__ug9",
      "name": "BRA Neighborhood Statistical Areas, Boston",
      "geocode_type_slug": "boston-administrative-geography",
      "geocode_type": "Boston, Administrative Geography",
      "attribute": {
        "attribute": "BOSNA_R_ID",
        "type": "xsd:double"
      },
      "abstract": "BRA Neighborhood Statistical Areas 2015, Boston. Provided by_
↪ [truncated for example]",
      "title": "BRA Neighborhood Statistical Areas 2015, Boston (BARI)",
      "expected_format": {
        "expected_zero_padded_length": -1,
        "is_zero_padded": false,
        "description": "Boston, Administrative Geography, Boston BRA Neighborhood_
↪ Statistical Area ID (integer). Examples: 1, 2, 3, ... 68, 69",
        "name": "Boston BRA Neighborhood Statistical Area ID (integer)"
      },
      "year": 2015,
      "id": 17
    }
  ],
  "success": true
}

```

## How Geoconnect Uses Join Target Information

When a user attempts to map a tabular file, the application looks in the Geoconnect database for `JoinTargetInformation`. If this information is more than 10 minutes\* old, the application will retrieve fresh information and save it to the db.

(\* Change the timing via the Django settings variable `JOIN_TARGET_UPDATE_TIME`.)

This `JoinTarget` info is used to populate HTML forms used to match a tabular file column to a `JoinTarget` column. Once a `JoinTarget` is chosen, the `JoinTarget ID` is an essential piece of information used to make an API call to the `WorldMap` and attempt to map the file.

## Retrieving Join Target Information from WorldMap API

The `get_join_targets()` function in `dataverse_layer_services.py` uses the WorldMap API, retrieves a list of available tabular file JointTargets. (See the `dataverse_layer_services` code in [GitHub](#).)

## Saving Join Target Information to Geoconnect Database

The `get_latest_jointtarget_information()` in `utils.py` retrieves recent JoinTarget Information from the database. (See the `utils` code in [GitHub](#).)

## Setting Up WorldMap Test Data

For the dataset page, this script gives a query to add test WorldMap map data. After the query is run, the “Explore Map” button should appear for a tabular file or shapefile. In the example SQL queries below, substitute `$DATASET_ID` and `$DATAFILE_ID` with the appropriate ID’s.

To add sample map data for a tabular file:

```
INSERT INTO maplayermetadata (id, isjoinlayer, joindescription, embedmaplink, ↵
↵ layerlink, layername, mapimagelink, worldmapusername, dataset_id, datafile_id)
VALUES (DEFAULT, true, 'This file was joined with WorldMap layer x, y, z',
'https://worldmap.harvard.edu/maps/embed/?layer=geonode:zip_codes_2015_zip_s9i',
↵ 'https://worldmap.harvard.edu/data/geonode:zip_codes_2015_zip_s9i',
'geonode:zip_codes_2015_zip_s9i',
'http://worldmap.harvard.edu/download/wms/27289/png?layers=geonode%3Azip_codes_2015_
↵ zip_s9i&#38;width=865&#38;bbox=-71.1911091251%2C42.2270382738%2C-70.9228275369%2C42.
↵ 3976144794&#38;service=WMS&#38;format=image%2Fpng&#38;srs=EPSG%3A4326&#38;
↵ request=GetMap&#38;height=550',
'admin', $DATASET_ID, $DATAFILE_ID));
```

To add sample map data for a tabular shapefile:

```
INSERT INTO maplayermetadata (id, isjoinlayer, embedmaplink, layerlink, layername, ↵
↵ mapimagelink, worldmapusername, dataset_id, datafile_id)
VALUES (DEFAULT, false,
'https://worldmap.harvard.edu/maps/embed/?layer=geonode:zip_codes_2015_zip_s9i',
↵ 'https://worldmap.harvard.edu/data/geonode:zip_codes_2015_zip_s9i',
'geonode:zip_codes_2015_zip_s9i',
'http://worldmap.harvard.edu/download/wms/27289/png?layers=geonode%3Azip_codes_2015_
↵ zip_s9i&#38;width=865&#38;bbox=-71.1911091251%2C42.2270382738%2C-70.9228275369%2C42.
↵ 3976144794&#38;service=WMS&#38;format=image%2Fpng&#38;srs=EPSG%3A4326&#38;
↵ request=GetMap&#38;height=550',
'admin', $DATASET_ID, $DATAFILE_ID);
```

---

Previous: *Universal Numerical Fingerprint (UNF)* | Next: *Shibboleth and OAuth*

## 5.21 SELinux

**Contents:**



- *Introduction*
- *Development Environment*
- *Recreating the shibboleth.te File*
  - *Ensure that SELinux is Enforcing*
  - *Removing the Existing shibboleth.te Rules*
  - *Exercising SELinux denials*
  - *Stub out the new shibboleth.te file*
  - *Iteratively Use audit2allow to Add Rules and Test Your Change*

### 5.21.1 Introduction

The `shibboleth.te` file below that is mentioned in the *Shibboleth* section of the Installation Guide was created on CentOS 6 as part of <https://github.com/IQSS/dataverse/issues/3406> but may need to be revised for future versions of RHEL/CentOS (pull requests welcome!). The file is versioned with the docs and can be found in the following location:

```
doc/sphinx-guides/source/_static/installation/files/etc/selinux/targeted/src/
policy/domains/misc/shibboleth.te
```

```
module shibboleth 1.0;

require {
    class file {open read};
    class sock_file write;
    class unix_stream_socket connectto;
    type httpd_t;
    type initrc_t;
    type var_run_t;
    type var_t;
}

allow httpd_t initrc_t:unix_stream_socket connectto;
allow httpd_t var_run_t:sock_file write;
allow httpd_t var_t:file {open read};
```

This document is something of a survival guide for anyone who is tasked with updating this file.

### 5.21.2 Development Environment

In order to work on the `shibboleth.te` file you need to `ssh` into a RHEL or CentOS box running Shibboleth (instructions are in the *Shibboleth* section of the Installation Guide) such as <https://beta.dataverse.org> or <https://demo.dataverse.org> that has all the commands below installed. As of this writing, the `polycoreutils-python` RPM was required.

### 5.21.3 Recreating the shibboleth.te File

If you're reading this page because someone has reported that Shibboleth doesn't work with SELinux anymore (due to an operating system upgrade, perhaps) you *could* start with the existing `shibboleth.te` file, but it is recommended that you create a new one instead to ensure that extra lines aren't included that are no longer necessary.

The file you're recreating is called a Type Enforcement (TE) file, and you can read more about it at [https://access.redhat.com/documentation/en-US/Red\\_Hat\\_Enterprise\\_Linux/6/html/Security-Enhanced\\_Linux/chap-Security-Enhanced\\_Linux-SELinux\\_Contexts.html](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/chap-Security-Enhanced_Linux-SELinux_Contexts.html)

The following doc may or may not be helpful to orient you: [https://access.redhat.com/documentation/en-US/Red\\_Hat\\_Enterprise\\_Linux/6/html/Security-Enhanced\\_Linux/sect-Security-Enhanced\\_Linux-Fixing\\_Problems-Allowing\\_Access\\_audit2allow.html](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/sect-Security-Enhanced_Linux-Fixing_Problems-Allowing_Access_audit2allow.html)

### Ensure that SELinux is Enforcing

If `getenforce` returns anything other than `Enforcing`, run `setenforce Enforcing` or otherwise configure SELinux by editing `/etc/selinux/config` and rebooting until SELinux is enforcing.

### Removing the Existing shibboleth.te Rules

Use `semodule -l | grep shibboleth` to see if the `shibboleth.te` rules are already installed. Run `semodule -r shibboleth` to remove the module, if necessary. Now we're at square one (no custom rules) and ready to generate a new `shibboleth.te` file.

### Exercising SELinux denials

As of this writing, there are two optional components of Dataverse that are known not to work with SELinux out of the box with SELinux: Shibboleth and rApache.

We will be exercising SELinux denials with Shibboleth, and the SELinux-related issues are expected out the box:

- Problems with the dropdown of institutions being created on the Login Page (“Internal Error - Failed to download metadata from /Shibboleth.sso/DiscoFeed.”).
- Problems with the return trip after you've logged into HarvardKey or whatever (“shibsp::ListenerException” and “Cannot connect to shibd process, a site administrator should be notified.”).

In short, all you need to do is try to log in with Shibboleth and you'll see problems associated with SELinux being enabled.

### Stub out the new shibboleth.te file

Iterate on the new `shibboleth.te` file wherever you like, such as the root user's home directory in the example below. Start by adding a `module` line like this:

```
echo 'module shibboleth 1.0;' > /root/shibboleth.te
```

Note that a version is required and perhaps it should be changed, but we'll stick with `1.0` for now. The point is that the `shibboleth.te` file must begin with that “`module`” line or else the `checkmodule` command you'll need to run later will fail. Your file should look like this:

```
module shibboleth 1.0;
# require lines go here
# allow lines go here
```

## Iteratively Use `audit2allow` to Add Rules and Test Your Change

Now that `shibboleth.te` has been stubbed out, we will iteratively add lines to it from the output of piping SELinux Access Vector Cache (AVC) denial messages to `audit2allow -r`. These errors are found in `/var/log/audit/audit.log` so tail the file as you attempt to log in to Shibboleth.

```
# tail -f /var/log/audit/audit.log | fgrep type=AVC
```

You should see messages that look something like this:

```
type=AVC msg=audit(1476728970.378:271405): avc: denied { write }
for pid=28548 comm="httpd" name="shibd.sock" dev=dm-2 ino=393300
scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:var_run_t:s0
tclass=sock_file
```

Next, pipe these message to `audit2allow -r` like this:

```
echo 'type=AVC msg=audit(1476728970.378:271405): avc: denied { write
} for pid=28548 comm="httpd" name="shibd.sock" dev=dm-2 ino=393300
scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:var_run_t:s0
tclass=sock_file' | audit2allow -r
```

This will produce output like this:

```
require {
    type var_run_t;
    type httpd_t;
    class sock_file write;
}

#===== httpd_t =====
allow httpd_t var_run_t:sock_file write;
```

Copy and paste this output into the `shibboleth.te` file you stubbed out above. Then, use the same `checkmodule`, `semodule_package`, and `semodule` commands documented in the *Shibboleth* section of the Installation Guide on your file to activate the SELinux rules you're constructing.

Once your updated SELinux rules are in place, try logging in with Shibboleth again. You should see a different AVC error. Pipe that error into `audit2allow -r` as well and put the resulting content into the `shibboleth.te` file you're constructing. As you do this, manually reformat the file using the following rules:

- Put the `require` block at the top.
- Within the `require` block, sort the lines.
- Put the `allow` lines at the bottom and sort them.
- Where possible, avoid duplicate lines by combining operations such as `open` and `read` into `{open read}`.
- Remove all comment lines.

Keep iterating until it works and then create a pull request based on your updated file. Good luck!

Many thanks to Bill Horka from IQSS for his assistance in explaining how to construct a SELinux Type Enforcement (TE) file!

---

Previous: [Geospatial Data](#)

## 5.22 Big Data Support

Big data support is highly experimental. Eventually this content will move to the Installation Guide.

### Contents:

- *Data Capture Module (DCM)*
  - *Install a DCM*
  - *Downloading rsync scripts via Dataverse API*
  - *How a DCM reports checksum success or failure to Dataverse*
  - *Steps to set up a DCM mock for Development*
  - *Troubleshooting*
  - *Steps to set up a DCM via Docker for Development*
    - \* *Docker Image Set-up*
    - \* *Optional steps for setting up the S3 Docker DCM Variant*
    - \* *Using the DCM Docker Containers*
    - \* *Additional DCM docker development tips*
- *Repository Storage Abstraction Layer (RSAL)*
  - *Steps to set up a DCM via Docker for Development*
    - \* *Using the RSAL Docker Containers*
  - *Configuring the RSAL Mock*
  - *Configuring download via rsync*

Various components need to be installed and configured for big data support.

### 5.22.1 Data Capture Module (DCM)

Data Capture Module (DCM) is an experimental component that allows users to upload large datasets via rsync over ssh.

#### Install a DCM

Installation instructions can be found at <https://github.com/sbgrid/data-capture-module/blob/master/doc/installation.md>. Note that shared storage (posix or AWS S3) between Dataverse and your DCM is required. You cannot use a DCM with Swift at this point in time.

Once you have installed a DCM, you will need to configure two database settings on the Dataverse side. These settings are documented in the *Configuration* section of the Installation Guide:

- `:DataCaptureModuleUrl` should be set to the URL of a DCM you installed.
- `:UploadMethods` should include `dcm/rsync+ssh`.

This will allow your Dataverse installation to communicate with your DCM, so that Dataverse can download rsync scripts for your users.

## Downloading rsync scripts via Dataverse API

The rsync script can be downloaded from Dataverse via API using an authorized API token. In the curl example below, substitute `$PERSISTENT_ID` with a DOI or Handle:

```
curl -H "X-Dataverse-key: $API_TOKEN" $DV_BASE_URL/api/datasets/
:persistentId/dataCaptureModule/rsync?persistentId=$PERSISTENT_ID
```

## How a DCM reports checksum success or failure to Dataverse

Once the user uploads files to a DCM, that DCM will perform checksum validation and report to Dataverse the results of that validation. The DCM must be configured to pass the API token of a superuser. The implementation details, which are subject to change, are below.

The JSON that a DCM sends to Dataverse on successful checksum validation looks something like the contents of `checksumValidationSuccess.json` below:

```
{
  "status": "validation passed",
  "uploadFolder": "OS708Y",
  "totalSize": 72
}
```

- `status` - The valid strings to send are `validation passed` and `validation failed`.
- `uploadFolder` - This is the directory on disk where Dataverse should attempt to find the files that a DCM has moved into place. There should always be a `files.sha` file and a least one data file. `files.sha` is a manifest of all the data files and their checksums. The `uploadFolder` directory is inside the directory where data is stored for the dataset and may have the same name as the “identifier” of the persistent id (DOI or Handle). For example, you would send `"uploadFolder": "DNXV2H"` in the JSON file when the absolute path to this directory is `/usr/local/glassfish4/glassfish/domains/domain1/files/10.5072/FK2/DNXV2H/DNXV2H`.
- `totalSize` - Dataverse will use this value to represent the total size in bytes of all the files in the “package” that’s created. If 360 data files and one `files.sha` manifest file are in the `uploadFolder`, this value is the sum of the 360 data files.

Here’s the syntax for sending the JSON.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST -H 'Content-type: application/
json' --upload-file checksumValidationSuccess.json $DV_BASE_URL/api/datasets/
:persistentId/dataCaptureModule/checksumValidation?persistentId=$PERSISTENT_ID
```

## Steps to set up a DCM mock for Development

See instructions at <https://github.com/sbgrid/data-capture-module/blob/master/doc/mock.md>

Add Dataverse settings to use mock (same as using DCM, noted above):

- `curl http://localhost:8080/api/admin/settings/:DataCaptureModuleUrl -X PUT -d "http://localhost:5000"`
- `curl http://localhost:8080/api/admin/settings/:UploadMethods -X PUT -d "dcm/rsync+ssh"`

At this point you should be able to download a placeholder rsync script. Dataverse is then waiting for news from the DCM about if checksum validation has succeeded or not. First, you have to put files in place, which is usually the job

of the DCM. You should substitute “X1METO” for the “identifier” of the dataset you create. You must also use the proper path for where you store files in your dev environment.

- `mkdir /usr/local/glassfish4/glassfish/domains/domain1/files/10.5072/FK2/X1METO`
- `mkdir /usr/local/glassfish4/glassfish/domains/domain1/files/10.5072/FK2/X1METO/X1METO`
- `cd /usr/local/glassfish4/glassfish/domains/domain1/files/10.5072/FK2/X1METO/X1METO`
- `echo "hello" > file1.txt`
- `shasum file1.txt > files.sha`

Now the files are in place and you need to send JSON to Dataverse with a success or failure message as described above. Make a copy of `doc/sphinx-guides/source/_static/installation/files/root/big-data-support/checksumValidationSuccess.json` and put the identifier in place such as “X1METO” under “uploadFolder”). Then use curl as described above to send the JSON.

### Troubleshooting

The following low level command should only be used when troubleshooting the “import” code a DCM uses but is documented here for completeness.

```
curl -H "X-Dataverse-key: $API_TOKEN" -X POST "$DV_BASE_URL/api/batch/jobs/import/datasets/files/$DATASET_DB_ID?uploadFolder=$UPLOAD_FOLDER&totalSize=$TOTAL_SIZE"
```

### Steps to set up a DCM via Docker for Development

If you need a fully operating DCM client for development purposes, these steps will guide you to setting one up. This includes steps to set up the DCM on S3 variant.

#### Docker Image Set-up

See <https://github.com/IQSS/dataverse/blob/develop/conf/docker-dcm/readme.md>

- Install docker if you do not have it

#### Optional steps for setting up the S3 Docker DCM Variant

- Before: the default bucket for DCM to hold files in S3 is named test-dcm. It is coded into `post_upload_s3.bash` (line 30). Change to a different bucket if needed.
  - Add AWS bucket info to `dcmsrv` - Add AWS credentials to `~/aws/credentials`
    - \* [default]
    - \* aws\_access\_key\_id =
    - \* aws\_secret\_access\_key =
- Dataverse configuration (on `dvsrv`):
  - Set S3 as the storage driver
    - \* `cd /opt/glassfish4/bin/`

- ```
* ./asadmin delete-jvm-options "\-Ddataverse.files.
storage-driver-id=file"

* ./asadmin create-jvm-options "\-Ddataverse.files.
storage-driver-id=s3"
```
- Add AWS bucket info to Dataverse - Add AWS credentials to ~/.aws/credentials

```
* [default]
* aws_access_key_id =
* aws_secret_access_key =
* Also: set region in ~/.aws/config to create a region file. Add these contents:
  · [default]
  · region = us-east-1
```
  - Add the S3 bucket names to Dataverse
    - \* S3 bucket for Dataverse

```
· /usr/local/glassfish4/glassfish/bin/asadmin create-jvm-options
  "\-Ddataverse.files.s3-bucket-name=igsstestdcmbucket"
```
    - \* S3 bucket for DCM (as Dataverse needs to do the copy over)

```
· /usr/local/glassfish4/glassfish/bin/asadmin create-jvm-options
  "\-Ddataverse.files.dcm-s3-bucket-name=test-dcm"
```
  - Set download method to be HTTP, as DCM downloads through S3 are over this protocol

```
curl -X PUT "http://localhost:8080/api/admin/settings/:DownloadMethods" -d
"native/http"
```

## Using the DCM Docker Containers

For using these commands, you will need to connect to the shell prompt inside various containers (e.g. `docker exec -it dvsrv /bin/bash`)

- Create a dataset and download rsync upload script
  - connect to client container: `docker exec -it dcm_client bash`
  - create dataset: `cd /mnt ; ./create.bash` ; this will echo the database ID to stdout
  - download transfer script: `./get_transfer.bash $database_id_from_create_script`
  - execute the transfer script: `bash ./upload-${database_id_from_create_script}.bash` , and follow instructions from script.
- Run script
  - e.g. `bash ./upload-3.bash` (3 being the database id from earlier commands in this example).
- Manually run post upload script on dcmsrv
  - for posix implementation: `docker exec -it dcmsrv /opt/dcm/scn/post_upload.bash`
  - for S3 implementation: `docker exec -it dcmsrv /opt/dcm/scn/post_upload_s3.bash`

## Additional DCM docker development tips

- You can completely blow away all the docker images with these commands (including non DCM ones!) -  
`docker-compose -f docmer-compose.yml down -v`
- There are a few logs to tail
  - `dvsrv : tail -n 2000 -f /opt/glassfish4/glassfish/domains/domain1/logs/server.log`
  - `dcmsrv:tail -n 2000 -f /var/log/lighttpd/breakage.log`
  - `dcmsrv:tail -n 2000 -f /var/log/lighttpd/access.log`
- You may have to restart the glassfish domain occasionally to deal with memory filling up. If deployment is getting realllllly slow, its a good time.

## 5.22.2 Repository Storage Abstraction Layer (RSAL)

### Steps to set up a DCM via Docker for Development

See <https://github.com/IQSS/dataverse/blob/develop/conf/docker-dcm/readme.md>

### Using the RSAL Docker Containers

- Create a dataset (either with the procedure mentioned in DCM Docker Containers, or another process)
- Publish the dataset (from the client container):  
`cd /mnt; ./publish_major.bash ${database_id}`
- Run the RSAL component of the workflow (from the host):  
`docker exec -it rsalsrv /opt/rsal/scn/pub.py`
- If desired, from the client container you can download the dataset following the instructions in the dataset access section of the dataset page.

### Configuring the RSAL Mock

Info for configuring the RSAL Mock: <https://github.com/sbgrid/rsal/tree/master/mocks>

Also, to configure Dataverse to use the new workflow you must do the following (see also the *Workflows* section):

1. Configure the RSAL URL:

```
curl -X PUT -d 'http://<myipaddr>:5050' http://localhost:8080/api/admin/settings/:RepositoryStorageAbstractionLayerUrl
```

2. Update workflow json with correct URL information:

Edit `internal-httpSR-workflow.json` and replace `url` and `rollbackUrl` to be the url of your RSAL mock.

3. Create the workflow:

```
curl http://localhost:8080/api/admin/workflows -X POST --data-binary @internal-httpSR-workflow.json -H "Content-type: application/json"
```

4. List available workflows:

```
curl http://localhost:8080/api/admin/workflows
```



5. Set the workflow (id) as the default workflow for the appropriate trigger:

```
curl http://localhost:8080/api/admin/workflows/default/PrePublishDataset -X
PUT -d 2
```

6. Check that the trigger has the appropriate default workflow set:

```
curl http://localhost:8080/api/admin/workflows/default/PrePublishDataset
```

7. Add RSAL to whitelist

8. When finished testing, unset the workflow:

```
curl -X DELETE http://localhost:8080/api/admin/workflows/default/
PrePublishDataset
```

## Configuring download via rsync

In order to see the rsync URLs, you must run this command:

```
curl -X PUT -d 'rsal/rsync' http://localhost:8080/api/admin/settings/
:DownloadMethods
```

To specify replication sites that appear in rsync URLs:

Download `add-storage-site.json` and adjust it to meet your needs. The file should look something like this:

```
{
  "hostname": "dataverse.librascholar.edu",
  "name": "LibraScholar, USA",
  "primaryStorage": true,
  "transferProtocols": "rsync,posix,globus"
}
```

Then add the storage site using curl:

```
curl -H "Content-type:application/json" -X POST http://localhost:8080/api/
admin/storageSites --upload-file add-storage-site.json
```

You make a storage site the primary site by passing “true”. Pass “false” to make it not the primary site. (id “1” in the example):

```
curl -X PUT -d true http://localhost:8080/api/admin/storageSites/1/
primaryStorage
```

You can delete a storage site like this (id “1” in the example):

```
curl -X DELETE http://localhost:8080/api/admin/storageSites/1
```

You can view a single storage site like this: (id “1” in the example):

```
curl http://localhost:8080/api/admin/storageSites/1
```

You can view all storage site like this:

```
curl http://localhost:8080/api/admin/storageSites
```

In the GUI, this is called “Local Access”. It’s where you can compute on files on your cluster.

```
curl http://localhost:8080/api/admin/settings/:LocalDataAccessPath -X PUT -d
"/programs/datagrid"
```

## 5.23 Workflows

Dataverse has a flexible workflow mechanism that can be used to trigger actions before and after Dataset publication.

**Contents:**

- *Introduction*
  - *Administration*
  - *Available Steps*
    - \* *log*
    - \* *pause*
    - \* *http/sr*
    - \* *archiver*

### 5.23.1 Introduction

Dataverse can perform two sequences of actions when datasets are published: one prior to publishing (marked by a `PrePublishDataset` trigger), and one after the publication has succeeded (`PostPublishDataset`). The pre-publish workflow is useful for having an external system prepare a dataset for being publicly accessed (a possibly lengthy activity that requires moving files around, uploading videos to a streaming server, etc.), or to start an approval process. A post-publish workflow might be used for sending notifications about the newly published dataset.

Workflow steps are created using *step providers*. Dataverse ships with an internal step provider that offers some basic functionality, and with the ability to load 3rd party step providers. This allows installations to implement functionality they need without changing the Dataverse source code.

Steps can be internal (say, writing some data to the log) or external. External steps involve Dataverse sending a request to an external system, and waiting for the system to reply. The wait period is arbitrary, and so allows the external system unbounded operation time. This is useful, e.g., for steps that require human intervention, such as manual approval of a dataset publication.

The external system reports the step result back to dataverse, by sending a HTTP POST command to `api/workflows/{invocation-id}`. The body of the request is passed to the paused step for further processing.

If a step in a workflow fails, Dataverse make an effort to roll back all the steps that preceded it. Some actions, such as writing to the log, cannot be rolled back. If such an action has a public external effect (e.g. send an EMail to a mailing list) it is advisable to put it in the post-release workflow.

---

**Tip:** For invoking external systems using a REST api, Dataverse's internal step provider offers a step for sending and receiving customizable HTTP requests. It's called *http/sr*, and is detailed below.

---

### Administration

A Dataverse instance stores a set of workflows in its database. Workflows can be managed using the `api/admin/workflows/` endpoints of the *Native API*. Sample workflow files are available in `scripts/api/data/workflows`.

At the moment, defining a workflow for each trigger is done for the entire instance, using the endpoint `api/admin/workflows/default/<trigger type>`.

In order to prevent unauthorized resuming of workflows, Dataverse maintains a “white list” of IP addresses from which resume requests are honored. This list is maintained using the `/api/admin/workflows/ip-whitelist` endpoint of the *Native API*. By default, Dataverse honors resume requests from localhost only (`127.0.0.1; ::1`), so set-ups that use a single server work with no additional configuration.

## Available Steps

Dataverse has an internal step provider, whose id is `:internal`. It offers the following steps:

### log

A step that writes data about the current workflow invocation to the instance log. It also writes the messages in its `parameters` map.

```
{
  "provider": ":internal",
  "stepType": "log",
  "parameters": {
    "aMessage": "message content",
    "anotherMessage": "message content, too"
  }
}
```

### pause

A step that pauses the workflow. The workflow is paused until a POST request is sent to `/api/workflows/{invocation-id}`.

```
{
  "provider": ":internal",
  "stepType": "pause"
}
```

### http/sr

A step that sends a HTTP request to an external system, and then waits for a response. The response has to match a regular expression specified in the step parameters. The url, content type, and message body can use data from the workflow context, using a simple markup language. This step has specific parameters for rollback.

```
{
  "provider": ":internal",
  "stepType": "http/sr",
  "parameters": {
    "url": "http://localhost:5050/dump/${invocationId}",
    "method": "POST",
    "contentType": "text/plain",
    "body": "START RELEASE ${dataset.id} as ${dataset.displayName}",
    "expectedResponse": "OK.*",
    "rollbackUrl": "http://localhost:5050/dump/${invocationId}",
  }
}
```

```
    "rollbackMethod": "DELETE ${dataset.id}"
  }
}
```

Available variables are:

- invocationId
- dataset.id
- dataset.identifier
- dataset.globalId
- dataset.displayName
- dataset.citation
- minorVersion
- majorVersion
- releaseStatus

### archiver

A step that sends an archival copy of a Dataset Version to a configured archiver, e.g. the DuraCloud interface of Chronopolis. See the [DuraCloud/Chronopolis Integration documentation](#) for further detail.

Note - the example step includes two settings required for any archiver and three (DuraCloud\*) that are specific to DuraCloud.

```
{
  "provider": ":internal",
  "stepType": "archiver",
  "parameters": {
    "stepName": "archive submission"
  },
  "requiredSettings": {
    ":ArchiverClassName": "string",
    ":ArchiverSettings": "string",
    ":DuraCloudHost": "string",
    ":DuraCloudPort": "string",
    ":DuraCloudContext": "string"
  }
}
```

## STYLE GUIDE

This style guide is meant to help developers implement clear and appropriate UI elements consistent with the Dataverse Project's standards.

### Contents:

## 6.1 Foundations

Foundation elements are the very basic building blocks to create a page in Dataverse. Here we will outline how we've applied Bootstrap CSS to our UI, and how the CSS settings in our stylesheet mesh with it. Each section includes links to relevant parts of the official Bootstrap guides and other useful resources, where you can find more detailed documentation. We will also outline other UI resources like FontCustom and Socicon and how they are utilized.

### Contents:

- *Grid Layout*
- *Typography*
- *Color Palette*
  - *Brand Colors*
  - *Text Colors*
  - *Link Colors*
  - *Contextual Classes*
- *Icons*
  - *Bootstrap Glyphicons*
  - *FontCustom Icon Font*
  - *Socicon Icon Font*
- *Logos*

### 6.1.1 Grid Layout

Bootstrap provides a responsive, fluid, 12-column grid system that we use to organize our page layouts.

We use the fixed-width `.container` class which provides responsive widths (i.e. auto, 750px, 970px or 1170px) based on media queries for the page layout, with a series of rows and columns for the content.

The grid layout uses `.col-sm-*` classes for horizontal groups of columns, inside a containing element with a `.row` class. Content should be placed within columns, and only columns may be immediate children of rows.

```
<div class="container">
  <div class="row">
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
    <div class="col-sm-1">.col-sm-1</div>
  </div>
  <div class="row">
    <div class="col-sm-8">.col-sm-8</div>
    <div class="col-sm-4">.col-sm-4</div>
  </div>
</div>
```

## 6.1.2 Typography

The typeface, text size, and line-height are set in the [Bootstrap CSS](#). We use Bootstrap’s global default `font-size` of **14px**, with a `line-height` of **1.428**, which is applied to the `<body>` and all paragraphs.

```
/* bootstrap.css */
body {
  font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
  font-size: 14px;
  line-height: 1.42857143;
}
```

## 6.1.3 Color Palette

The default color palette is set in the [Bootstrap CSS](#). It provides the background, border, text and link colors used across the application.

### Brand Colors

Dataverse uses a particular color palette to help users quickly and easily identify the different types of objects: dataverses, datasets, and files.

We use our brand color, a custom burnt orange `{color:#C55B28;}`, which is set in our CSS stylesheet, “`structure.css`”. There is also a set of blue “dataset” classes and grey “file” classes, used to help identify those objects when searching and navigating the application.

```
/* structure.css */
.bg-dataverse {
  background: #C55B28;
}
```

```
.bg-dataset {
  background: #337AB7;
}
.bg-file {
  background: #F5F5F5;
}

#navbarFixed .navbar-brand {
  color: #C55B28;
}
#navbarFixed .icon-dataverse {
  color: #C55B28;
}
```

```
<div class="bg-dataverse">...</div>
<div class="bg-dataset">...</div>
<div class="bg-file">...</div>
```

```
/* structure.css */
.text-dataverse {
  color: #C55B28;
}
.text-dataset {
  color: #31708F;
}
.text-file {
  color: #F5F5F5;
}
```

```
<p class="text-dataverse">...</p>
<p class="text-dataset">...</p>
<p class="text-file">...</p>
```

## Text Colors

Text color is the default setting from Bootstrap CSS.

```
/* bootstrap.css */
body {
  color: #333;
}
```

```
<p>...</p>
```

## Link Colors

Link color is the default setting from Bootstrap CSS. The hover state color is set to 15% darker.

**Please note**, there is a CSS override issue with the link color due to the use of both a Bootstrap stylesheet and a PrimeFaces stylesheet in the UI. We've added CSS such as `.ui-widget-content {color: #428BCA;}` to our stylesheet to keep the link color consistent.

```
/* bootstrap.css */
a {
  color: #337AB7;
}
a:hover {
  color: #23527C;
}

/* structure.css */
.ui-widget-content a {
  color: #337AB7;
}
.ui-widget-content a:hover, .ui-widget-content a:focus {
  color: #23527C;
}
```

```
<a>...</a>
```

## Contextual Classes

Contextual classes from Bootstrap CSS can be used to style background and text colors. Semantic colors include various colors assigned to meaningful contextual values. We convey meaning through color with a handful of emphasis utility classes.

```
<div class="bg-primary">...</div>
<div class="bg-success">...</div>
<div class="bg-info">...</div>
<div class="bg-warning">...</div>
<div class="bg-danger">...</div>
```

```
<p class="text-muted">...</p>
<p class="text-primary">...</p>
<p class="text-success">...</p>
<p class="text-info">...</p>
<p class="text-warning">...</p>
<p class="text-danger">...</p>
```

## 6.1.4 Icons

We use various icons across the application, which we get from Bootstrap, FontCustom and Socicon. They appear in buttons, in message blocks or as default thumbnails for dataverses, datasets, and files.

### Bootstrap Glyphicons

There are over 250 glyphs in font format from the Glyphicon Halflings set provided by Bootstrap. We utilize these mainly as icons inside of buttons and in message blocks.

```
<span class="glyphicon glyphicon-search"></span>
<span class="glyphicon glyphicon-user"></span>
<span class="glyphicon glyphicon-lock"></span>
```



## FontCustom Icon Font

With the use of [FontCustom](#) we were able to generate our own custom icon webfonts. We use these in the search result cards to help distinguish between dataverse, dataset and file results.

```
<span class="icon-dataverse text-dataverse"></span>
<span class="icon-dataset text-dataset"></span>
<span class="icon-file text-file"></span>
```

## Socicon Icon Font

We use [Socicon](#) for our custom social icons. In the footer we use icons for Twitter and Github. In our Share feature, we also use custom social icons to allow users to select from a list of social media channels.

```
<span class="socicon socicon-github" title="Dataverse On GitHub"></span>
<span class="socicon socicon-twitter" title="Dataverse On Twitter"></span>
<span class="socicon socicon-facebook" title="Dataverse On Facebook"></span>
```

## 6.1.5 Logos

The Dataverse Project logo (below) is displayed in the footer, and was the basis for the creation of the application’s icons and favicon.

Create both print and web version of the Dataverse Project logo by downloading this vector-based SVG file: `dataverse_project_logo.svg`

The brand logo (below) was created as a custom icon to represent the concept of a “dataverse.” It is used as the brand logo in the Bootstrap navbar component and across the application.

Create both print and web version of the Dataverse brand logo by downloading this vector-based SVG file: `Dataverse_brand_icon.svg`

## 6.2 Patterns

Patterns are what emerge when using the foundation elements together with basic objects like buttons and alerts, more complex Javascript components from [Bootstrap](#) like tooltips and dropdowns, and AJAX components from [PrimeFaces](#) like datatables and commandlinks.

### Contents:

- *Navbar*
- *Breadcrumbs*
- *Tables*
- *Forms*
- *Buttons*
  - *Action Buttons*
  - *Form Buttons*

– *Icon-Only Buttons*

- *Pagination*
- *Labels*
- *Alerts*

– *Message Classes*

- *Images*
- *Panels*
- *Tabs*
- *Modals*

## 6.2.1 Navbar

The `Navbar` component from Bootstrap spans the top of the application and contains the logo/branding, aligned to the left, plus search form and links, aligned to the right.

When logged in, the account name is a dropdown menu, linking the user to account-specific content and the log out link.

```
<nav id="navbarFixed" class="navbar navbar-default navbar-fixed-top">
  <div class="container">
    <div class="navbar-header">
      <a href="#" onclick="return false;">
        <span class="navbar-brand"><i id="icon-dataverse" class="icon-dataverse"></i>
↳ Dataverse</span>
      </a>
    </div>
    <div class="collapse navbar-collapse" id="topNavBar">
      <ul class="nav navbar-nav navbar-right">
        <li>
          ...
        </li>
      </ul>
    </div>
  </div>
</nav>
```

## 6.2.2 Breadcrumbs

The breadcrumbs are displayed under the header, and provide a trail of links for users to navigate the hierarchy of containing objects, from file to dataset to dataverse. It utilizes a JSF `repeat` component to iterate through the breadcrumbs.

```
<div id="breadcrumbNavBlock" class="container" jsf:rendered="{true}">
  <ui:repeat value="{page.breadcrumbs}" var="breadcrumb" varStatus="status">
    <h:outputText value=" " styleClass="breadcrumbCarrot" rendered="{true}"/>
    <div class="breadcrumbBlock">
      ...
    </div>
  </ui:repeat>
</div>
```

## 6.2.3 Tables

Most tables use the `DataTable` components from PrimeFaces and are styled using the `Tables` component from Bootstrap.

```
<p:dataTable id="itemTable" styleClass="headerless-table margin-top" value="#{page.
↪item}" var="item" widgetVar="itemTable">
  <p:column>
    ...
  </p:column>
</p:dataTable>
```

## 6.2.4 Forms

Forms fulfill various functions across the site, but we try to style them consistently. We use the `.form-horizontal` layout, which uses `.form-group` to create a grid of rows for the labels and inputs. The consistent style of forms is maintained using the `Forms` component from Bootstrap. Form elements like the `InputText` component from PrimeFaces are kept looking clean and consistent across each page.

```
<div class="form-horizontal">
  <div class="form-group">
    <label for="userNameEmail" class="col-sm-3 control-label">
      #{bundle['user.username']}
    </label>
    <div class="col-sm-4">
      <p:inputText id="userName" styleClass="form-control"></p>
    </div>
  </div>
</div>
```

Here are additional form elements that are common across many pages, including required asterisks, icon tooltips, placeholder text, input info message with popover link, and validation error message.

```
<div class="form-group form-col-container col-sm-9 edit-compound-field">
  <div class="form-col-container col-sm-12">
    <p class="help-block">
      <h:outputFormat value="#{bundle.htmlAllowedMsg}" escape="false">
        <f:param value="#{bundle.htmlAllowedTags}"/>
      </h:outputFormat>
    </p>
    <label class="control-label" for="metadata_#{subdsf.datasetFieldType.name}">
      #{subdsf.datasetFieldType.localeTitle}
      <h:outputText styleClass="glyphicon glyphicon-asterisk text-danger" value=
↪ "" />
      <span class="glyphicon glyphicon-question-sign tooltip-icon" data-toggle=
↪ "tooltip" data-placement="auto right" data-original-title="#{subdsf.
↪ datasetFieldType.localeDescription}"></span>
    </label>
    <div>
      <p:inputTextarea value="#{dsfv.valueForEdit}" id="description" tabIndex="#
↪ {block.index+1}" rows="5" cols="60" styleClass="form-control" />
      <div class="alert-danger" jsf:rendered="#{!empty subdsf.validationMessage}
↪ ">
```

```
        <strong>#{subdsf.validationMessage}</strong>
      </div>
    </div>
  </div>
</div>
```

## 6.2.5 Buttons

There are various types of buttons for various actions, so we have many components to use, including the `CommandButton` component and `CommandLink` component from PrimeFaces, as well as the basic JSF `Link` component and `OutputLink` component. Those are styled using the `Buttons` component, `Button Groups` component and `Buttons Dropdowns` component from Bootstrap.

### Action Buttons

For action buttons on a page, we include an icon and text label. Action buttons are generally aligned to the right side of the page.

```
<div class="btn-group" jsf:rendered="#{true}">
  <button type="button" id="editDataSet" class="btn btn-default dropdown-toggle" data-
  ↪toggle="dropdown">
    <span class="glyphicon glyphicon-pencil"/> Edit <span class="caret"></span>
  </button>
  <ul class="dropdown-menu pull-right text-left" role="menu">
    <li>
      <h:outputLink> ... </h:outputLink>
    </li>
    <li class="dropdown-submenu pull-left">
      <a tabindex="-1" href="#">Option</a>
      <ul class="dropdown-menu">
        <li>
          <h:link> ... </h:link>
        </li>
        <li>
          <h:link> ... </h:link>
        </li>
      </ul>
    </li>
    ...
  </ul>
</div>
```

### Form Buttons

Form buttons typically appear at the bottom of a form, aligned to the left. They do not have icons, just text labels. The primary button is styled differently.

```
<div class="button-block">
  <p:commandButton id="save" styleClass="btn btn-default" value="#{bundle.saveChanges}"
  ↪" action="#{page.save}" update="@form,:messagePanel" />
  <p:commandButton id="cancel" styleClass="btn btn-link" value="#{bundle.cancel}"
  ↪action="#{page.cancel}" process="@this" update="@form">
    <p:resetInput target="@form" />
  </p:commandButton>
</div>
```

```
</p:commandButton>
</div>
```

## Icon-Only Buttons

There are a few places where we use icon-only buttons with no text label. For these buttons, we do utilize tooltips that display on hover, containing a text label.

We use the style class `.no-text` with the `.glyphicon` class to fix spacing issues from margins and padding applied to buttons with text labels.

```
<p:commandLink styleClass="btn btn-default btn-sm bootstrap-button-tooltip" title="#"
→{bundle.add}" actionListener="#{Page.add(valCount.index + 1)}">
  <h:outputText styleClass="glyphicon glyphicon-plus no-text"/>
</p:commandLink>
<p:commandLink styleClass="btn btn-default btn-sm bootstrap-button-tooltip" title="#"
→{bundle.delete}" actionListener="#{Page.remove(valCount.index)}">
  <h:outputText styleClass="glyphicon glyphicon-minus no-text"/>
</p:commandLink>
```

## 6.2.6 Pagination

We use the `Pagination` component from `Bootstrap` for paging through search results.

```
<ul class="pagination">
  <li class="{include.page == '1' ? 'disabled' : ''}">
    <h:outputLink value="{page.page}">
      <h:outputText value="&#171;"/>
      ...
    </h:outputLink>
  </li>
  <li class="{include.page == '1' ? 'disabled' : ''}">
    <h:outputLink value="{page.page}">
      <h:outputText value="&lt; #{bundle.previous}"/>
      ...
    </h:outputLink>
  </li>
  ...
  <li class="{include.page == include.totalPages ? 'disabled' : ''}">
    <h:outputLink value="{page.page}">
      <h:outputText value="{bundle.next} &gt;"/>
      ...
    </h:outputLink>
  </li>
  <li class="{include.page == include.totalPages ? 'disabled' : ''}">
    <h:outputLink value="{page.page}">
      <h:outputText value="&#187;"/>
      ...
    </h:outputLink>
  </li>
</ul>
```

## 6.2.7 Labels

The **Labels** component from Bootstrap is used for publication status (DRAFT, In Review, Unpublished, Deaccessioned), and Dataset version, as well as Tabular Data Tags (Survey, Time Series, Panel, Event, Genomics, Network, Geospatial).

```
<span class="label label-default">Version 2.0</span>
<span class="label label-primary">DRAFT</span>
<span class="label label-success">In Review</span>
<span class="label label-info">Geospatial</span>
<span class="label label-warning">Unpublished</span>
<span class="label label-danger">Deaccessioned</span>
```

## 6.2.8 Alerts

For our help/information, success, warning, and error message blocks we use a custom built UI component based on the **Alerts** component from Bootstrap.

```
<div class="alert alert-success" role="alert">...</div>
<div class="alert alert-info" role="alert">...</div>
<div class="alert alert-warning" role="alert">...</div>
<div class="alert alert-danger" role="alert">...</div>
```

## Message Classes

Style classes can be added to p, div, span and other elements to add emphasis to inline message blocks.

```
<p class="help-block">
  <span class="text-muted">...</span>
</p>

<p class="help-block">
  <span class="glyphicon glyphicon-ok-sign text-success"></span> <span class="text-
  ↳ success">...</span>
</p>

<p class="help-block">
  <span class="glyphicon glyphicon-asterisk text-info"></span> <span class="text-info
  ↳ ">...</span>
</p>

<p class="help-block">
  <span class="glyphicon glyphicon-warning-sign text-warning"></span> <span class=
  ↳ "text-warning">...</span>
</p>

<p class="help-block">
  <span class="glyphicon glyphicon-exclamation-sign text-danger"></span> <span class=
  ↳ "text-danger">...</span>
</p>
```

## 6.2.9 Images

For images, we use the **GraphicImage** component from PrimeFaces, or the basic JSF **GraphicImage** component.

To display images in a responsive way, they are styled with `.img-responsive`, an [Images CSS class](#) from Bootstrap.

```
<p:graphicImage styleClass="img-responsive" value="{Page.imageId}?imageThumb=400" />
```

## 6.2.10 Panels

The most common of our containers, the [Panels component](#) from Bootstrap is used to add a border and padding around sections of content like metadata blocks. Displayed with a header and/or footer, it can also be used with the [Collapse plugin](#) from Bootstrap.

```
<div class="panel panel-default">
  <div class="panel-body">
    Basic panel example
  </div>
</div>

<div class="panel panel-default">
  <div data-toggle="collapse" data-target="#panelCollapse0" class="panel-heading">
    <span class="text-info">Panel Heading &#160;</span><span class="glyphicon glyphicon-
    ↪chevron-up"/></span>
  </div>
  <div id="panelCollapse0" class="panel-body form-horizontal collapse in">
    <div class="form-group">
      <label class="col-sm-4 control-label">
        Label
      </label>
      <div class="col-sm-6">
        Value
      </div>
    </div>
  </div>
</div>
```

## 6.2.11 Tabs

Tabs are used to provide content panes on a page that allow the user to view different sections of content without navigating to a different page.

We use the [TabView component](#) from PrimeFaces, which is styled using the [Tab component](#) from Bootstrap.

```
<p:tabView id="tabView" widgetVar="content" activeIndex="{Page.selectedTabIndex}">
  <p:ajax event="tabChange" listener="{Page.tabChanged}" update="@this" />
  <p:tab id="dataTab" title="{bundle.files}">
    ...
  </p:tab>
  ...
</p:tabView>
```

## 6.2.12 Modals

Modals are dialog prompts that act as popup overlays, but don't create a new browser window. We use them for confirmation on a delete to make sure the user is aware of the consequences of their actions. We also use them to allow users to execute simple actions on a page without requiring them to navigate to and from a separate page.

Buttons usually provide the UI prompt. A user clicks the button, which then opens a [Dialog component](#) or [Confirm Dialog component](#) from PrimeFaces that displays the modal with the necessary information and actions to take.

The modal is styled using the [Modal component](#) from Bootstrap, for a popup window that prompts a user for information, with overlay and a backdrop, then header, content, and buttons. We can use style classes from Bootstrap for large (`.bs-example-modal-lg`) and small (`.bs-example-modal-sm`) width options.

```
<!-- Large modal -->
<button type="button" class="btn btn-primary" data-toggle="modal" data-target=".bs-
↪example-modal-lg">Large modal</button>

<div class="modal bs-example-modal-lg" tabindex="-1" role="dialog" aria-labelledby=
↪"myLargeModalLabel">
  <div class="modal-dialog modal-lg" role="document">
    <div class="modal-content">
      ...
    </div>
  </div>
</div>
```



## HOW THE GUIDES ARE ORGANIZED

The guides are documentation that explain how to use Dataverse, which are divided into the following sections: User Guide, Installation Guide, Developer Guide, API Guide and Style Guide. The User Guide is further divided into primary activities: finding & using data, adding Datasets, administering dataverses or Datasets, and Dataset exploration/visualizations. Details on all of the above tasks can be found in the Users Guide. The Installation Guide is for people or organizations who want to host their own Dataverse. The Developer Guide contains instructions for people who want to contribute to the Open Source Dataverse project or who want to modify the code to suit their own needs. Finally, the API Guide is for Developers that work on other applications and are interested in connecting with Dataverse through our APIs.



## OTHER RESOURCES

### **Dataverse Project Site**

Additional information about the Dataverse Project itself including presentations, information about upcoming releases, data management and citation, and announcements can be found at <http://dataverse.org/>

### **User Group**

As the user community grows we encourage people to share ideas, ask questions, or offer suggestions for improvement. Go to <https://groups.google.com/group/dataverse-community> to register to our dataverse community group.

### **Follow Us on Twitter**

For up to date news, information and developments, follow our twitter account: <https://twitter.com/dataverseorg>

### **Support**

We maintain an email based support service that is free of charge. We attempt to respond within one business day to all questions and if it cannot be resolved immediately, we will let you know what to expect.

The support email address is [support@dataverse.org](mailto:support@dataverse.org).

### **Reporting Issues and Contributing**

Report bugs and add feature requests in [GitHub Issues](#) or use [GitHub pull requests](#), if you have some code, scripts or documentation that you'd like to share. If you have a **security issue** to report, please email [security@dataverse.org](mailto:security@dataverse.org).



## INDICES AND TABLES

- genindex
- modindex
- search