Dataverse Documentation

Release 4.2.3

Dataverse Team

December 04, 2015

Contents

These guides are for the most recent version of Dataverse. For the guides for **version 4.2.2** please go here.

CHAPTER 1

User Guide

Contents:

1.1 Account Creation & Management

As a registered user, you can:

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

1.1.1 Create User Account

1. In the top right corner of each page, click on the Sign Up link.
2. Fill out all the fields, then click the Create Account button at the end. 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). Congrats you now have a Dataverse account!

1.1.2 Edit Your 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.

1.1.3 Generate Your API Token

1. To generate your API token, click on your name in the hearder on right hand side and then click on Account Information.
2. On the top right of your account page, click on the "Edit Account" button and click on API Token in the list.
3. Your API Token is located on that page.

1.1.4 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 access to edit. You are able to filter through all the dataverses, datasets, and files listed there using the filter box or use the facets on the left side to only view a specific Publication Status or Role.

1.1.5 Notifications: Setup & Maintainance

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

Dataverse will email your unread notifications once a day. Notifications will only be emailed one time even if you haven't read the notification on the Dataverse site.

1.1.6 Reset Your Account Password

If you cannot remember the password for your Dataverse account, click on Log In in the top right corner of any page. Once on that page, click on the Forgot Password? link below where you would enter your username and password. Enter your email address and click Submit Password Request to receive an email with a link to reset your password.

*Note: if you have forgotten your username, you can do this same process to receive your username in an email.

1.2 Finding and Using 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 a 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 the dataverses, datasets, and files within a particular dataverse by using the search bar found on a dataverse page. For example, if you are on the Murray Research Archive Dataverse page, you can search that specific dataverse's contents by using the search bar and/or facets displayed on the page.

Basic Search

From the Dataverse homepage, you can begin searching with an exact phrase search or entering a search term or query in the search box that says, "Search this dataverse"

Search results features

- **Facets: to the left of the search results, there are several facets a user can click on to narrow the number of results displa**
  - Choosing a facet: to choose a facet, click on the facet to choose it

  - Removing a facet: to remove a facet, click on the X next to a chosen facet in search string above the results OR a chosen facet can be removed by clicking on the X next to it in the facets pane to the left of the results
  - Viewing more or less: the top five results show in each facet, to view more, click on "More..." in the bottom right of a facet. 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, results cards that match your term or query appear underneath the s**

  - Relevancy of results: each results 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), newest or oldest, or 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 you to be able to return to at a later time.

Advanced Search

In an advanced search, you can refine your criteria by choosing which Metadata fields to search. You can perform an advanced search on Citation metadata fields as well as domain specific metadata fields depending on the metadata selected for your Dataverse (Social Sciences & Humanities, Geospatial Astronomy & Astrophysics, and Life Sciences). Additionally, you can perform an advanced search for dataverses and files.

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 happens when a user hasn't entered a search term or query into the basic search bar. Browsing is the default for a user when they are on the Dataverse homepage or 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.

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. Below that information, the files, metadata, and version information for the dataset are available.

Download Citation

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

Download Files

Within the Files tab on a dataset page, a user can either Explore tabular data files using TwoRavens, Download All File Formats + Information or individually download the Original File Format, Tab Delimited Format, Variable Metadata, Data File Citation (RIS Format or EndNote XML), or Subset (options appear depending on file format).

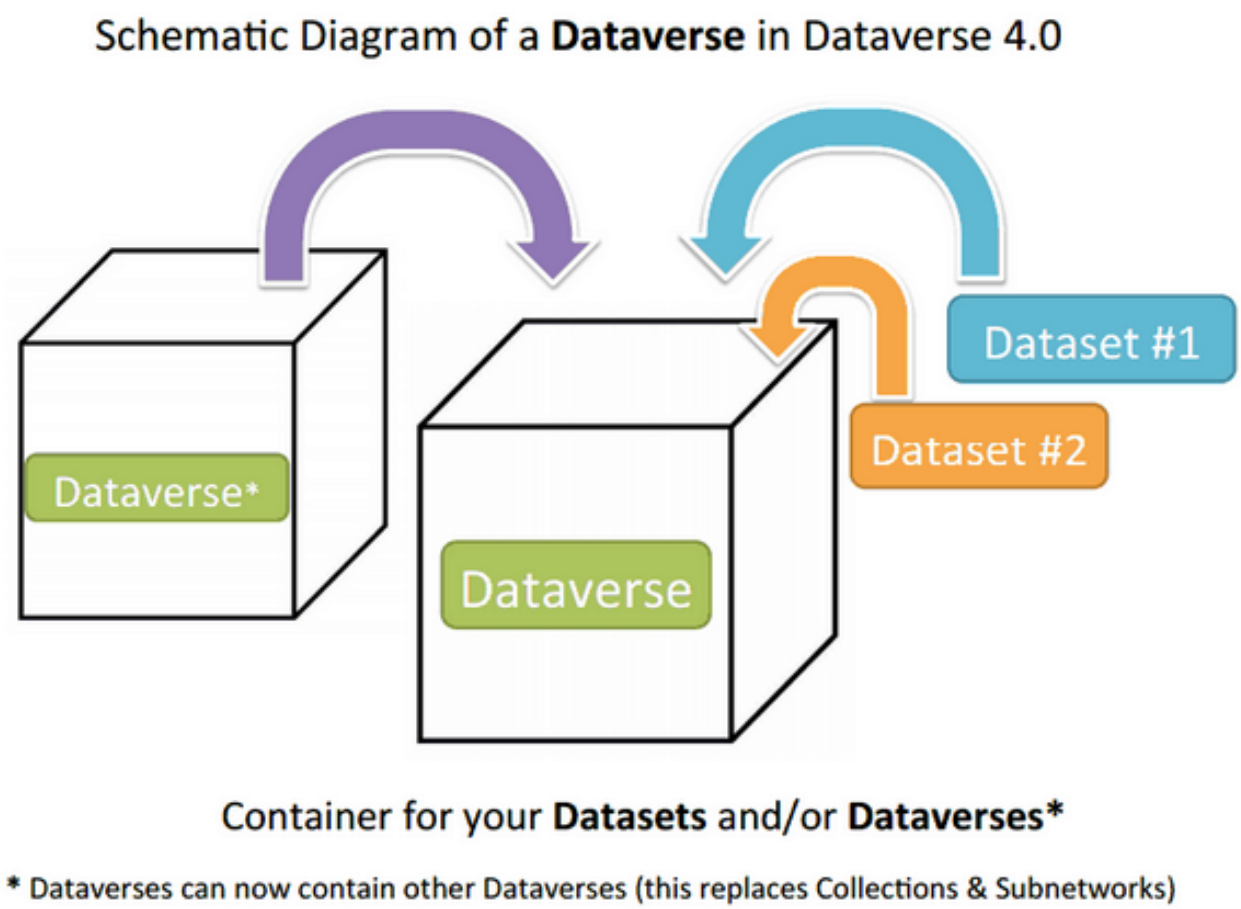
To download more than one file at a time, select the files you would like to download and then click the Download button above the files. The selected files will download in zip format.

Explore Data

Please see the Data Exploration Guide.

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.



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.

1.3.1 Create a Dataverse (within the "root" Dataverse)

Creating a dataverse is easy but first you must be a registered user (see Create Account).

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 the Dataverse URL field, the URL for your Dataverse changes (http//.../'url'), which affects links to this 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 to 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 home page of your dataverse and in the search result list. The description field supports certain HTML tags (<a>, <b>, <blockquote>,
, <code>, <del>, <dd>, <dl>, <dt>, , <hr>, <h1>-<h3>, <i>, <img>, <kbd>, <li>, <ol>, <p>, <pre>, <s>, <sup>, <sub>, , <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 & Institution.
- **Choose the sets of Metadata Elements 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 (insert link here)
- **Select facets for this dataverse**: 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. The facets are simply metadata fields that can be used to help others easily find dataverses and datasets within this dataverse. You can select as many facets as you would like.

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 homepage 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 + Widgets* : upload a logo for your dataverse, add a link to your department or personal website, and select colors for your dataverse in order to brand it. Also, you can 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

1.3.3 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 metadafields 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.

1.3.4 Theme + Widgets

The Theme + Widgets feature provides you with a way to customize the look of your dataverse as well as provide code for you to put on your personal website to have your dataverse appear there. (For adding your dataverse to an OpenScholar site, follow these instructions.)

For Theme, you can decide either to use the customization from the dataverse above yours or upload your own image file. Supported image types are jpeg, tiff, or png and should be no larger than 500kb. The maximum display for an image file in a dataverse's theme is 940 piels wide by 120 pixels high. Additionally, you can select the colors for the header of your dataverse and the text that appears in your dataverse. You can also add a link to your personal website, the website for your organization or institution, your department, journal, etc.

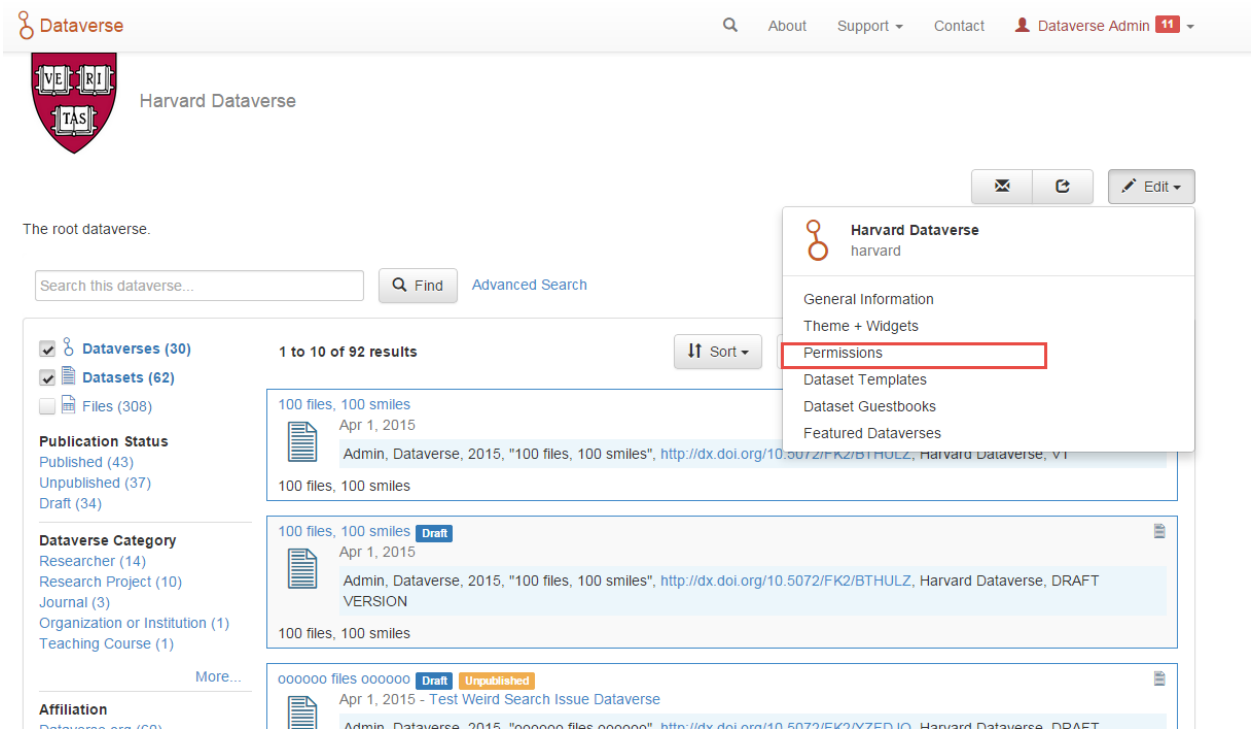
There are two options for Widgets, a Dataverse Search box widget and a Dataverse Listing widget. The Dataverse Search Box will add a search box to your website that when someone enters a search term in and clicks Find, will bring them to Dataverse to see the results. The Dataverse Listing widget will provide a listing of all your dataverses and datasets. When someone clicks on a dataverse or dataset in the widget, it will bring them to your dataverse to see the actual data. Within the Widgets page, you can copy and paste the code for the widget you would like to have on your website.

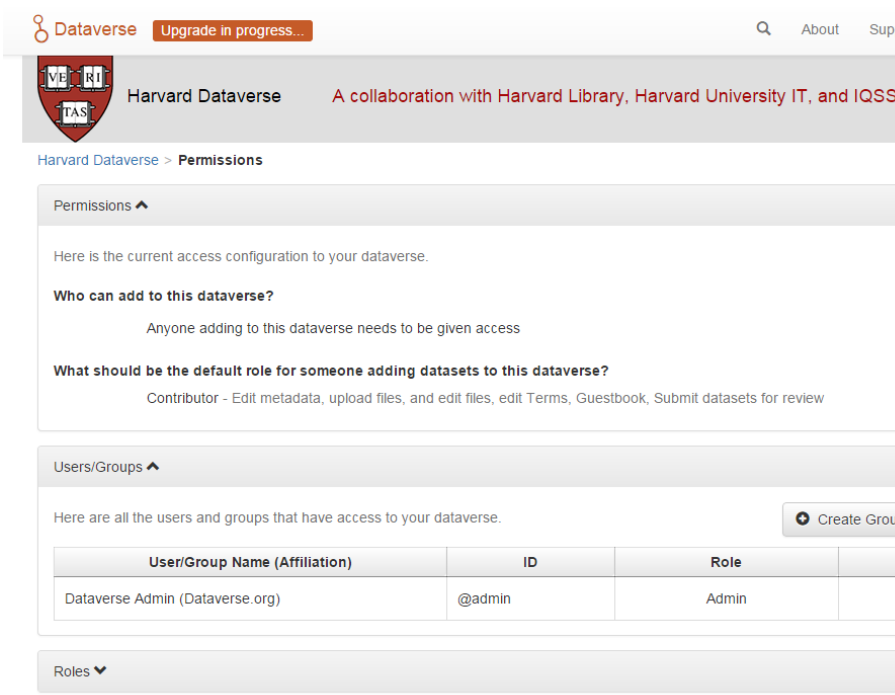
1.3.5 Adding Widgets to an OpenScholar Website

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 (we recommend using both), fill out the form, and then drag it to where you would like it to display in the page.

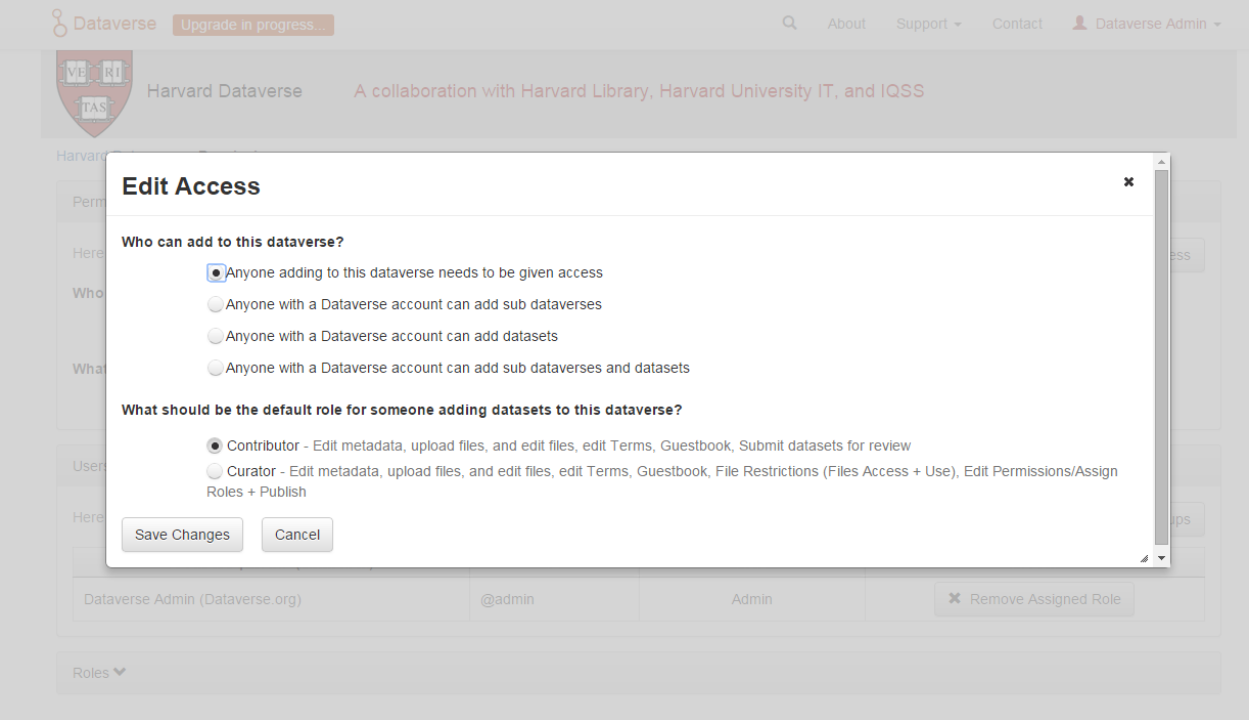
1.3.6 Permissions

When you access a dataverse's permissions page, you will see there are three sections: Permissions, Users/Groups, and Roles.





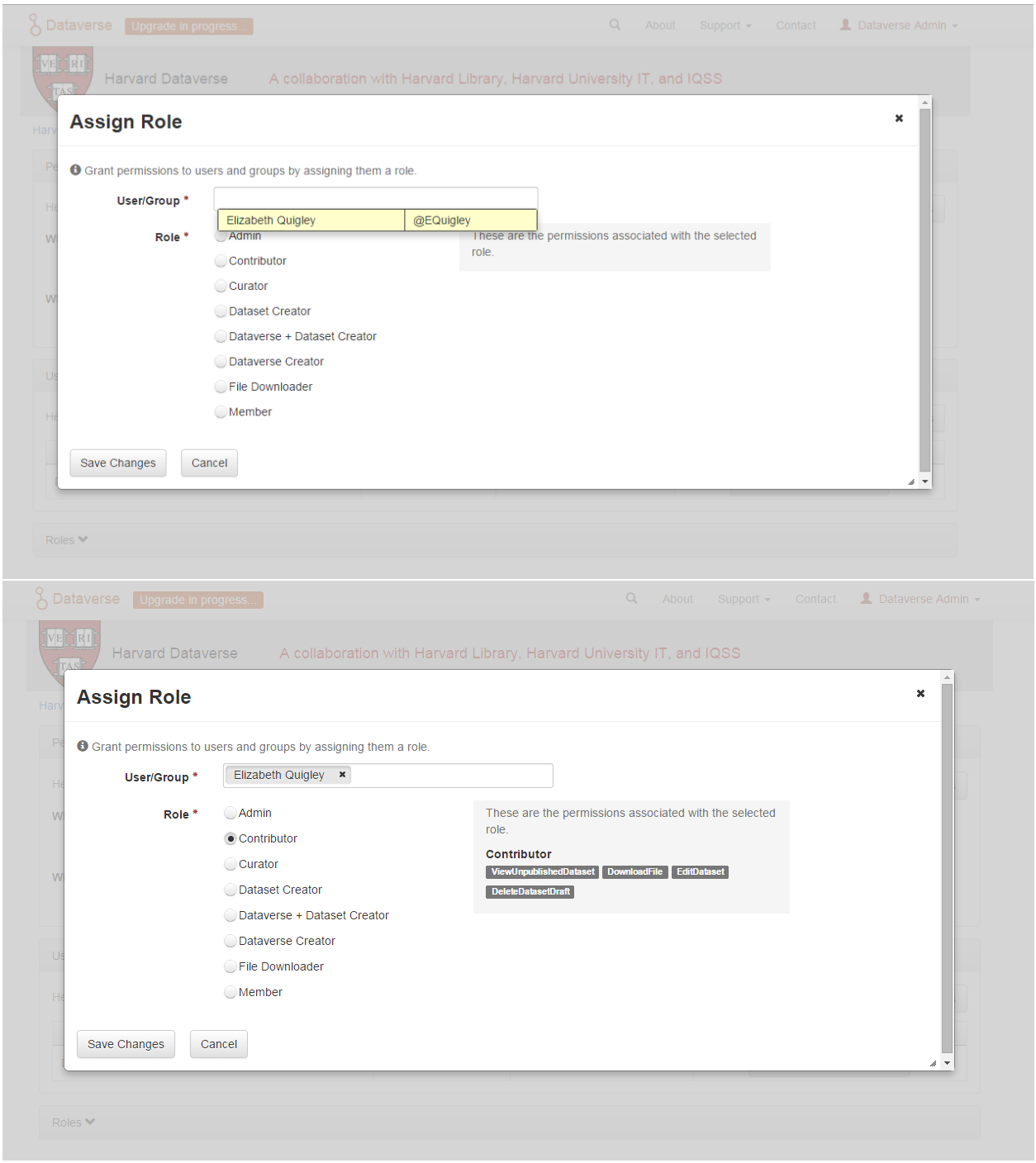
Clicking on Permissions will bring you to this page:
By clicking on the Edit Access button, you are able to change the settings allowing no one or anyone to add either dataverses or datasets to a dataverse.



The Edit Access pop up allows you to also select if someone adding a dataset to this dataverse should be allowed to publish it (Curator role) or if the dataset will be submitted to the administrator of this dataverse to be reviewed then published (Contributor role). These Access settings can be changed at any time.

Assigning a role

You can also give access to a Dataverse user to allow them to access an unpublished dataverse as well as other roles. To do this, click on the Assign Roles to Users/Groups button in the Users/Groups section. You can also give multiple users the same role at one time.



This roles can be removed at any time.

1.3.7 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 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 similiar 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.

1.3.8 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 rightside 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.

1.3.9 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.

1.3.10 Linked Dataverses + Linked Datasets

Currently, the ability to link a dataverse to another dataverse or a dataset to a dataverse is a super user only feature.

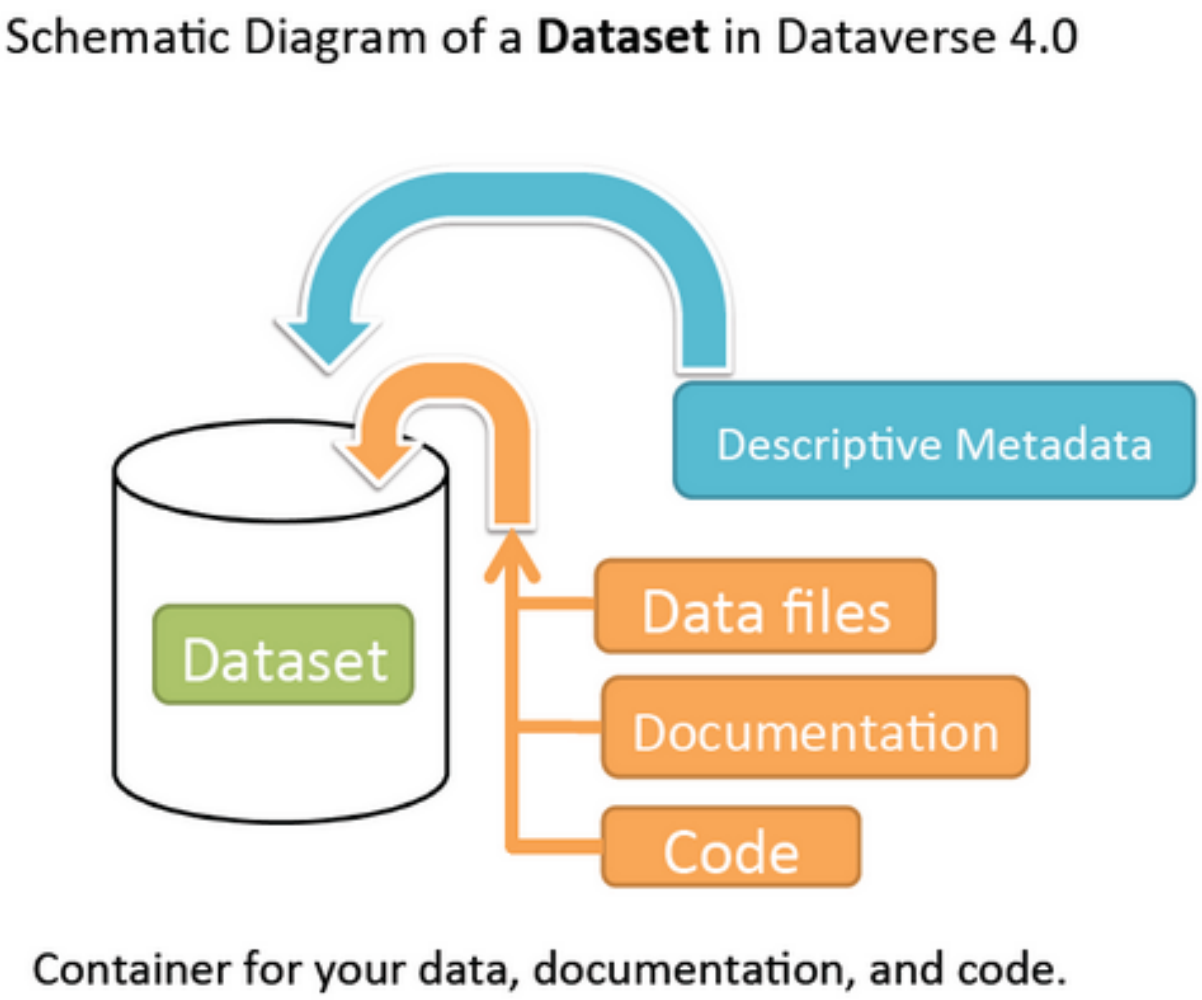
If you need to have a dataverse or dataset linked in the Harvard Dataverse installation, please contact support@dataverse.org.

1.3.11 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.



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 - see *File Handling and Uploading* section below for more details).

For more details about what Citation and Domain specific metadata is supported please see our Appendix.

1.4.2 File Handling + Uploading

All file formats are supported, up to 2GB per file for the Harvard Dataverse. Please contact support@dataverse.org if you need to upload a file that is larger than 2GB. You can also add descriptions and categorize each of them by adding tags.

The file types listed below are supported by additional functionality, which can include downloading in different formats, subsets, file-level metadata preservation, file-level data citation; and exploration through data visualization and analysis.

Tabular Data Files

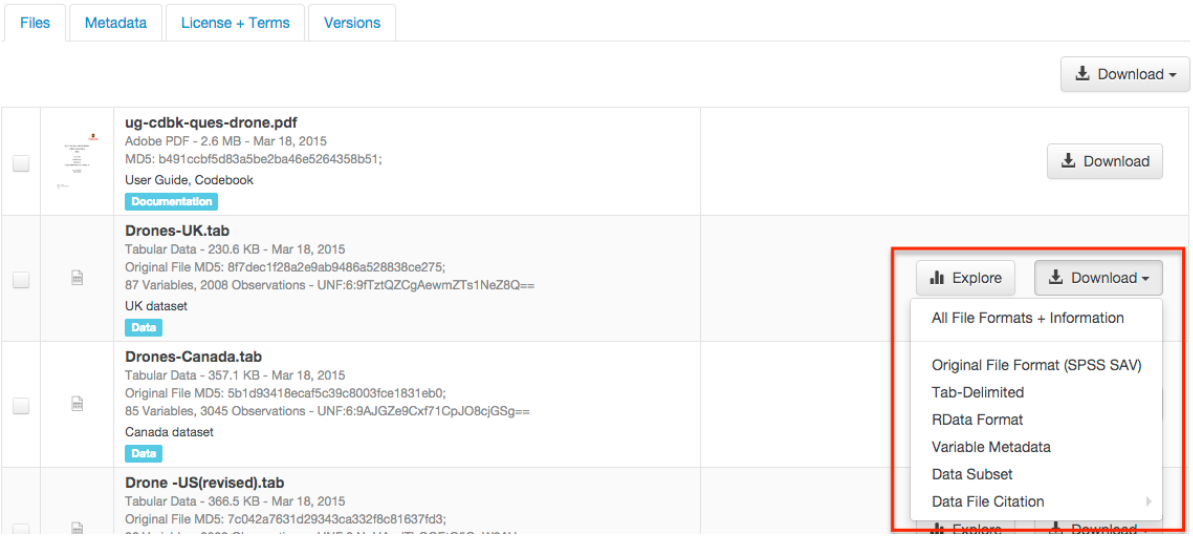
Files in certain formats - Stata, SPSS, R, Excel(xlsx) and CSV - may be ingested as tabular data (see "Tabular Data Ingest" section for details). Tabular data files can be further explored and manipulated with TwoRavens - a statistical data exploration application integrated with Dataverse. It allows 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). To download subsets of variables click on the "Download" button found next to a relevant tabular file and select "Data Subset" in the dropdown menu. You will then be able to create your subset using the interface opened in a new window (this functionality is also provided by the TwoRavens project). See the TwoRavens documentation section for more information.

To use the ingest functionality for tabular files, a file can only be up to 2GB in size, however, to upload a tabular file without using ingest, a file can be up to 2GB in size.

To use the ingest functionality for RData files, a file can only be up to 1MB in size, however, to upload a RData file without using ingest, a file can be up to 2GB in size.

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 or EndNote XML 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. 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 it 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), you will get an error message and the file will uploads as is.

Support for unpacking tar files will be added when this ticket is closed: https://github.com/IQSS/dataverse/issues/2195.

Advanced Options

There are several advanced options available for certain file types.

- Image files: jpgs, pngs, and tiff 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.

1.4.3 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 to get a Data Citation with a DOI (e.g., the Dataset Title, Author, Description, Contact Email and Subject).
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 1: 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 meatdata fields (i.e., Description) : <a>, <b>, <blockquote>,
, <code>, <del>, <dd>, <dl>, <dt>, , <hr>, <h1>-<h3>, <i>, <img>, <kbd>, <li>, <ol>, <p>, <pre>, <s>, <sup>, <sub>, , <strike>, <ul>.

1.4.4 Edit Dataset

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 on the Edit button above the files and 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

All of these actions, besides editing file metadata, will happen within this page and not bring you to another page. If you restrict files, you will also be asked to fill out the Terms of Access for the files. If Terms of Access already exist, you will be asked to confirm them.

1.4.5 Upload New Files

To upload new files to a dataset, go the dataset you want to update and click on the Upload Files Button in the files tab. From there you will be brought to the Upload page for the dataset. Once you have uploaded files, you will be able to edit the file metadata, restrict, add tags, or delete them before saving.

1.4.6 Terms

In the Terms tab, which can also be found by clicking on the Edit dropdown button of a Dataset, you can setup 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 Waiver + Dataset Terms of Use

Starting with Dataverse version 4.0, all new datasets will default to a CC0 public domain dedication . CC0 facilitates reuse and extensibility of research data. Our Community Norms as well as good scientific practices expect that proper credit is given via citation. If you are unable to give your datasets a CC0 waiver you may enter your own custom Terms of Use for your Datasets.

* **Legal Disclaimer:** these Community Norms are not a substitute for the CC0 waiver or custom terms and licenses applicable to each dataset. Please be advised that 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.

Setting up Custom Terms of Use for Datasets

If you are unable to use a CC0 waiver for your datasets you are able to set your own custom terms of use. To do so, select "No, do not apply CC0 - "Public Domain Dedication" and a Terms of Use textbox 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, click on "Additional Information [+]".

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, click on "Additional Information [+]".

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 Dataverse Management Guide > Dataset Guestbook section.

1.4.7 Permissions

Dataset-Level

Dataset permissions are located under Permissions in the Edit button on a dataset page. The dataset permissions page has two sections: Users/Groups and Roles.

To give someone access to view your unpublished dataset or edit your published or unpublished dataset, click on the Assign Roles to Users/Groups button in the Users/Groups section.

File-Level

If you have restricted specific files the file-level permissions is where you will need to go to grant users/groups access to specific restricted files. Dataset file permissions are located under Permissions in the Edit button on a dataset page. The file permissions page has two sections: Users/Groups and Files.

To give someone access to your restricted files, click on the Grant Access to Users/Groups button in the Users/Groups section.

1.4.8 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.

1.4.9 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.

1.4.10 Dataset Versioning

Versioning is important for long term-research data management where metadata and/or files are updated over time.

Once you have published a dataset, any metadata or file changes (e.g, by uploading a new file, changing file metadata, adding or editing metadata) will be tracked in our versioning feature. For example if you were at version 1 of your dataset, and you edit your dataset a new draft version of this dataset will be created. To get to the already published version 1 of your dataset, click on the "View Dataset Versions" button on the top left section of your dataset. To go back to the unpublished version click on the same button. Once you are ready 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, and depending on the types of changes you have made, you will be asked to select 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 (example: if you were at 1.0 you will go to 2.0).

Dataset Versions Tab

To view what has exactly changed starting from the originally published version to any subsequent published versions: click on 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 (can be version 1 and a draft), you can click the Show Details link in the Versions tab to learn more about the metadata fields and files that were either added or edited.

If you have more than two versions of a dataset, you can select any two versions to compare the differences between them. After selecting two versions, click on the "Show Differences" button to see the version differences details.

1.4.11 Deaccession Your Dataset [not recommended]

Deaccessioning a dataset or a version of a dataset 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 a dataset you've already published (or add a new one and publish it), click on Edit Dataset, then 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 from a dropdown list of options. There is also a free-text box to add more details as to why this was deaccessioned. 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 are able to go in and create a new draft for the dataset. For example, you have a version 1 and version 2 of a dataset, both published, and deaccession version 2. You are then able to edit version 1 of the dataset and a new draft 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: (see the sections below for more information on each of the supported 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)

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.

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 idea 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.

[TODO: a brief explanation of the DataVariable and related objects? A link to a more technical documentation writeup in the developers guide?]

1.5.3 SPSS

SPSS data files (POR and SAV formats).

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 know 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 R Data Format

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

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 DVN.

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 DVN.

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 DVN 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 DVN, 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 DVN; 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,"tzone")<-"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 DVN 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 DVN 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.

1.5.5 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.

1.5.6 CSV

Ingest of Comma-Separated Values files as tabular data.

Dataverse will make an attempt to turn CSV files uploaded by the user into tabular data.

Main formatting requirements:

The first line must contan a comma-separated list of the variable names;

All the lines that follow must contain the same number of comma-separated values as the first, variable name line.

Limitations:

Except for the variable names supplied in the top line, very little information describing the data can be obtained from a CSV file. We strongly recommend using one of the supported rich files formats (Stata, SPSS and R) to provide more

descriptive metadata (informatinve lables, categorical values and labels, and more) that cannot be encoded in a CSV file.

The application will however make an attempt to recognize numeric, string and date/time values in CSV files.

Tab-delimited Data Files:

Tab-delimited files could be ingested by replacing the TABs with commas.

1.6 Data Exploration Guide

Contents:

1.6.1 TwoRavens: Tabular Data Exploration

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 sola Fide 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 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.

Scenario Example:

1.6.2 WorldMap: Geospatial Data Exploration

WorldMap

WorldMap is developed by the Center for Geographic Analysis (CGA) at Harvard and is an open source software that helps researchers visualize and explore their data in maps. The WorldMap and Dataverse collaboration allows researchers to be able to upload shapefiles to Dataverse for long term storage and receive a persistent identifier (through DOI) as well as be able to easily move into WorldMap to interact with the data and save to WorldMap as well. GeoConnect is the platform integrating Dataverse and WorldMap together and what you will use to visualize your data.

Uploading Shapefiles to Dataverse

To get started, you will need to create a dataset in Dataverse. For more detailed instructions on creating a dataset, read the Dataset + File Management portion of this user guide.

Dataverse recognizes ZIP files that contain the components of a shapefile and will ingest them as a ZIP.

Once you have uploaded your ZIP files comprising a shapefile, a Map Data button will appear next to the file in the dataset.

Mapping your data with Geoconnect

In order to use the WorldMap and Dataverse integration, your dataset will need to be published. Once it has been published, you will be able to use the MapData button. Click on the Map Data button to be brought to GeoConnect, the portal between Dataverse and WorldMap that will process your shapefile and send it to WorldMap.

To get started with visualizing your shapefile, click on the blue Visualize on WorldMap button in GeoConnect. It may take 30 seconds or longer for the data to be sent to WorldMap and then back to GeoConnect

Once the visualizing has finished, you will be able to style your map through Attribute, Classification Method, Number of Intervals, and Colors. At any time, you can view the map on WorldMap if you would like to see how it will be displayed there.

After styling your map, you can delete it or return to Dataverse. If you decide to delete the map, it will no longer appear on WorldMap. By returning to Dataverse, you will send the styled map layer to WorldMap as well as to Dataverse where a preview will be available of the map layer you styled using GeoConnect.

To map the shapefile again, all you will need to do is click the Map Data button again.

1.7 Appendix

Additional documentation complementary to the User Guide.

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 639-2 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).

CHAPTER 2

Installation Guide

Contents:

2.1 Prerequisites

2.1.1 Java

Oracle JDK or OpenJDK 1.7.x. Use the latest available. MacOS X comes with the JDK (but make sure you keep it updated). On a RedHat and similar Linux distributions, install it with something like

```
$ yum install java-1.7.0-openjdk-devel
```

2.1.2 Glassfish

Glassfish Version 4.1 is required.

Important: once Glassfish is installed, a new version of the WELD library (v2.2.10.SP1) must be downloaded and installed. This fixes a serious issue in the library supplied with Glassfish 4.1.

- 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
```

- Remove the stock WELD jar; download WELD v2.2.10.SP1 and install it in the modules folder:

```
$ cd /usr/local/glassfish4/glassfish/modules
$ /bin/rm weld-osgi-bundle.jar
$ wget http://central.maven.org/maven2/org/jboss/weld/weld-osgi-bundle/2.2.10.SP1/weld-osgi-bund
$ /usr/local/glassfish4/bin/asadmin start-domain domain1
```

- Verify Weld version:

```
$ /usr/local/glassfish4/bin/asadmin osgi lb | grep 'Weld OSGi Bundle'
```

2.1.3 PostgreSQL

1. Installation

Version 9.3 is recommended.

1A. RedHat and similar systems:

We recommend installing Postgres from the EPEL repository:

```
$ wget http://yum.postgresql.org/9.3/redhat/rhel-6-x86_64/pgdg-centos93-9.3-1.noarch.rpm
rpm -ivh pgdg-centos93-9.3-1.noarch.rpm

$ yum install postgresql93-server.x86_64
$ chkconfig postgresql-9.3 on
$ service postgresql-9.3 initdb
$ service postgresql-9.3 start
```

1B. MacOS X:

A distribution from http://www.enterprisedb.com is recommended. Fink and MacPorts distributions are also readily available. See http://www.postgresql.org/download/macosx/ for more information.

2. Configure access to PostgresQL for the installer script

- The installer script needs to have direct access to the local PostgresQL server via Unix domain sockets. This is configured by the line that starts with "local all all" in the pg_hba.conf file. The location of this file may vary depending on the distribution. But if you followed the suggested installtion instructions above, it will be `/var/lib/pgsql/9.3/data/pg_hba.conf` on RedHat (and similar) and `/Library/PostgreSQL/9.3/data/pg_hba.conf` on MacOS. Make sure the line looks like this (it will likely be pre-configured like this already):

```
local all all       peer
```

- If the installer still fails to connect to the databse, we recommend changing this configuration entry to `trust`:

```
local all all      trust
```

  This is a security risk, as it opens your database to anyone with a shell on your server. It does not however compromise remote access to your system. Plus you only need this configuration in place to run the installer. After it's done, you can safely reset it to how it was configured before.

3. Configure database access for the Dataverse application

- The application will be talking to PostgresQL over TCP/IP, using password authentication. If you are running PostgresQL on the same server as Glassfish, we strongly recommend that you use the localhost interface to connect to the databse. Make you sure you accept the default value `localhost` when the installer asks you for the PostgresQL server address. Then 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 password
```

- 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 password
```

(`[ADDRESS]` should be the numeric IP address of the Glassfish server).

- 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: you must restart Postgres** for the configuration changes to take effect! On RedHat and similar (provided you installed Postgres as instructed above):

```
$ service postgresql-9.3 restart
```

(On MacOS, you may need to restart your system, to be sure).

2.1.4 Solr

- **Download and Install Solr::** $ wget https://archive.apache.org/dist/lucene/solr/4.6.0/solr-4.6.0.tgz $ tar xvzf solr-4.6.0.tgz $ rsync -auv solr-4.6.0 /usr/local/ $ cd /usr/local/solr-4.6.0/example/solr/collection1/conf/ $ mv schema.xml schema.xml.backup $ wget -q –no-check-certificate https://github.com/IQSS/dataverse/raw/master/conf/solr/4.6.0/schema.xml

In order to start Solr, you will need a customized schema file that is supplied in the Dataverse distribution bundle.

2.1.5 Start Up Scripts

- Example of Glassfish Startup file:

```
set -e
ASADMIN=/usr/local/glassfish4/bin/asadmin
case "$1" in
start)
        echo -n "Starting GlassFish server: glassfish"
        # Increase file descriptor limit:
        ulimit -n 32768
        # Allow "memory overcommit":
        # (basically, this allows to run exec() calls from inside the
        # app, without the Unix fork() call physically hogging 2X
        # the amount of memory glassfish is already using)
        echo 1 > /proc/sys/vm/overcommit_memory

        # Set UTF8 as the default encoding:
        LANG=en_US.UTF-8; export LANG
        $ASADMIN start-domain domain1
        echo "."
        ;;
          stop)
        echo -n "Stopping GlassFish server: glassfish"

        $ASADMIN stop-domain domain1
        echo "."
```

```
            ;;

              *)
            echo "Usage: /etc/init.d/glassfish {start|stop}"
            exit 1
            esac
    exit 0
```

2.2 Dataverse Application 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. At this point the installer only runs on RedHat 6.* derivatives.

2.2.1 Download and run the installer

Download the installer package (`dvnstall.zip`). Unpack the zip file - this will create the directory `dvinstall`. Execuite the installer script (`install`):

```
cd dvinstall
```

```
./install
```

The script will prompt you for some configuration values. If this is a test/evaluation installation, it should be safe to accept nthe defaults for most of the settiongs. For for a developer's installation we recommend that you choose `localhost` for the host name.

The script is to a large degree a derivative of the old installer from DVN 3.x. It is written in Perl.

All the Glassfish configuration tasks performed by the installer are isolated in the shell script `scripts/install/glassfish-setup.sh` (as `asadmin` commands).

2.3 Application Configuration

Much of the Dataverse Application configuration is done by the automated installer (described above). This section documents the additional configuration tasks that need to be done after you run the installer.

2.3.1 Dataverse Admin Account

Now that you've run the application installer and have your own Dataverse instance, you need to configure the Dataverse Administrator user. By default installer pre-sets the Admin credentials as follows:

```
First Name: Dataverse
Last Name:  Admin
Affiliation: Dataverse.org
Position: Admin
Email: dataverse@mailinator.com
```

Log in as the user dataverseAdmin with the password "admin" and change these values to suit your installation.

(Alteratively, you can modify the file `dvinstall/data/user-admin.json` in the installer bundle **before** you run the installer. The password is in `dvinstall/setup-all.sh`, which references this JSON file.)

2.3.2 Solr Configuration

Dataverse requires a specific Solr schema file called *schema.xml* that can be found in the Dataverse distribution. It should replace the default *example/solr/collection1/conf/schema.xml* file that ships with Solr.

If `WARN org.eclipse.jetty.http.HttpParser - HttpParser Full for /127.0.0.1:8983` appears in the Solr log, adding `<Set name="requestHeaderSize">8192</Set>` (or a higher number of bytes) to Solr's jetty.xml in the section matching the XPath expression `//Call[@name='addConnector']/Arg/New[@class='org.eclipse.jetty.server.bio.SocketConnector']` may resolve the issue. See also https://support.lucidworks.com/hc/en-us/articles/201424796-Error-when-submitting-large-query-strings-

Solr Security

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://wiki.apache.org/solr/SolrSecurity

2.3.3 Settings

ApplicationPrivacyPolicyUrl

Specify a URL where users can read your Privacy Policy, linked from the bottom of the page.

```
curl -X PUT -d http://best-practices.dataverse.org/harvard-policies/harvard-privacy-policy
http://localhost:8080/api/admin/settings/:ApplicationPrivacyPolicyUrl
```

ApplicationTermsOfUse

Upload a text file containing the Terms of Use to be displayed at sign up.

```
curl -X PUT -d@/tmp/apptou.html http://localhost:8080/api/admin/settings/:ApplicationTerms
```

ApiTermsOfUse

Upload a text file containing the API Terms of Use.

```
curl -X PUT -d@/tmp/api-tos.txt http://localhost:8080/api/admin/settings/:ApiTermsOfUse
```

SolrHostColonPort

Set `SolrHostColonPort` to override `localhost:8983`.

```
curl -X PUT -d localhost:8983 http://localhost:8080/api/admin/settings/:SolrHostColonPort
```

SearchHighlightFragmentSize

Set `SearchHighlightFragmentSize` to override the default value of 100 from https://wiki.apache.org/solr/HighlightingParameters#hl.fragsize

```
curl -X PUT -d 320 http://localhost:8080/api/admin/settings/:SearchHighlightFragmentSize
```

ShibEnabled

This setting is experimental per Shibboleth.

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.

```
curl -X PUT -d 2147483648 http://localhost:8080/api/admin/settings/:MaxFileUploadSizeInByte
```

GuidesBaseUrl

Set `GuidesBaseUrl` to override the default value "http://guides.dataverse.org".

```
curl -X PUT -d http://dataverse.example.edu http://localhost:8080/api/admin/settings/:Guide
```

2.3.4 JVM Options

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.foobar.edu, and not the less appealling server-123.socsci.foobar.edu to appear exclusively in all the registered global identifiers, Data Deposit API records, etc.

To change the option on the command line:

```
asadmin delete-jvm-options "-Ddataverse.fqdn=old.example.com"
```

```
asadmin create-jvm-options "-Ddataverse.fqdn=dataverse.example.com"
```

The `dataverse.fqdn` JVM option also affects the password reset feature.

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

and specify the alternative protocol and port number.
For example, configured in domain.xml:

```
<jvm-options>-Ddataverse.fqdn=dataverse.foobar.edu</jvm-options>
<jvm-options>-Ddataverse.siteUrl=http://${dataverse.fqdn}:8080</jvm-options>
```

dataverse.auth.password-reset-timeout-in-minutes

Set the `dataverse.auth.password-reset-timeout-in-minutes` option if you'd like to override the default value put into place by the installer.

2.3.5 Dropbox Configuration

- Add JVM option in the domain.xml:

```
asadmin create-jvm-options "-Ddataverse.dropbox.key=<Enter your dropbox key
here>"
```

The guide is intended for anyone who needs to install the Dataverse app.

If you encounter any problems during installation, please contact the development team at support@thedata.org or our Dataverse Users Community.

2.4 R, rApache and TwoRavens

Eventually, this document may be split into several parts, dedicated to individual components - such as R, rApache and the TwoRavens applications. Particularly, if the TwoRavens team creates an "official" distribution with their own installation manual.

2.4.1 0. PREREQUISITS

a. httpd (Apache):

```
yum install httpd
```

Disable SELinux on httpd:

```
setenforce permissive
```

```
getenforce
```

https strongly recommended; signed certificate (as opposed to self-signed) is recommended.

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:

yum install R R-devel

(EPEL distribution recommended; version 3.* required; 3.1.* recommended as of writing this)

c. rApache:

rpm distribution from the HMDC systems group is recommended (latest available version is 1.2.6, as of writing this). The rpm requires Apache libapreq2, that should be available via yum.

install rApache as follows:

```
yum install libapreq2
rpm -ivh http://mirror.hmdc.harvard.edu/HMDC-Public/RedHat-6/rapache-1.2.6-rpm0.x86_64.rpm
```

d. Install libcurl-devel:

(provides /usr/bin/curl-config, needed by some 3rd-party R packages; package installation *will fail silently* if it's not found!):

```
yum install libcurl-devel
```

Make sure you have the standard GNU compilers installed (needed for 3rd-party R packages to build themselves).

Update: As of Aug. 4 2015, it appears the following rpms had to be installed:

```
yum install openssl-devel yum install xml2-devel
```

Again, without these rpms, R package devtools was failing to install, silently or with a non-informative error message. Note: this package `devtools` has proven to be very flaky; it is being very actively maintained, new dependencies are being constantly added and new bugs introduced... however, it is only needed to install the package `Zelig`, the main R workhorse behind TwoRavens. It cannot be installed from CRAN, like all the other 3rd party packages we use - becase TwoRavens requires version 5, which is still in beta. So devtools is needed to build it from sources downloaded directly from github. Once Zelig 5 is released, we'll be able to drop the requirement for devtools - and that will make this process much simpler. For now, be prepared for it to be somewhat of an adventure.

2.4.2 1. Set Up R

R is used both by the Dataverse application, directly, and the TwoRavens companion app.

Two distinct interfaces are used to access R: Dataverse uses Rserve; and TwoRavens sends jobs to R running under rApache using Rook interface.

We provide a shell script (`conf/R/r-setup.sh` in the Dataverse source tree; you will need the other 3 files in that directory as well - https://github.com/IQSS/dataverse/conf/R/) that will attempt to install the required 3rd party packages; it will also configure Rserve and rserve user. rApache configuration will be addressed in its own section.

The script will attempt to download the packages from CRAN (or a mirror) and GitHub, so the system must have access to the internet. On a server fully firewalled from the world, packages can be installed from downloaded sources. This is left as an exercise for the reader. Consult the script for insight.

See the Appendix for the information on the specific packages, and versions that the script will attempt to install.

2.4.3 2. Install the TwoRavens Application

a. download the app

from https://github.com/IQSS/TwoRavens/archive/master.zip

b. unzip

and **rename the resulting directory** `dataexplore`. Place it in the web root directory of your apache server. We'll assume `/var/www/html/dataexplore` in the examples below.

c. run the installer

a scripted, interactive installer is provided at the top level of the TwoRavens distribution. Run it as:

```
cd /var/www/html/dataexplore
chmod +x install.pl
./install.pl
```

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`	
Apache host address	local hostname	rApache host
Apache host port	`80`	rApache port (**see the next section** for the discussion on ports!)
Apache web protocol	`http`	http or https for rApache (https recommended)
Dataverse URL	`http://{local hostname}:8080`	URL of the Dataverse from which TwoRavens will be receiving metadata and data files.

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

This URL **must** be configured in the settings of your Dataverse application! This can be done by issuing the following settings API call:

```
curl -X PUT -d {TWORAVENS_URL} http://localhost:8080/api/admin/settings/:TwoRavensUrl
```

where "{TWORAVENS_URL}" is the URL reported by the installer script (as in the example above).

d. Ports configuration

By default, Glassfish will install itself on ports 8080 and 8181 (for http and https, respectively), and Apache - on port 80 (the default port for http). Under this configuration, your Dataverse will be accessible at http://{your host}:8080 and https://{your host}:8181; 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 may be the easiest to set up if you are simply trying out/testing the 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 real users, you'll probably want to run Glassfish on ports 80 and 443. This way, there will be no non-standard ports in the Dataverse url visible to the users. Then you'll need to configure the Apache to run on some other port - for example, 8080, instead of 80. This port will only appear in the URL for the TwoRavens app. If you want to use this configuration - or any other that is not the default one described above! - it is your job to reconfigure Glassfish and Apache to run on the desired ports **before** you run the TwoRavens installer.

Furthermore, while the default setup assumes http as the default protocol for both the Dataverse and TwoRavens, https is strongly recommended for a real production system. Again, this will be your responsibility, to configure https in both Glassfish and Apache. Glassfih comes pre-configured to run https on port 8181, with a *self-signed certificiate*. For a production system, you will most certainly will want to obtain a properly signed certificate and configure Glassfish to use it. Apache does not use https out of the box at all. Again, it is the responsibility of the installing user, to configure Apache to run https, and, providing you are planning to run rApache on the same host as the Dataverse, use the same SSL certificate as your Glassfish instance. Again, it will need to be done before you run the installer script above. All of this may involve some non-trivial steps and will most likely require help from your local network administrator - unless you happen to be your local sysadmin. Unfortunately, we cannot provide step-by-step instructions for these tasks. As the actual steps required will likely depend on the specifics of how your institution obtains signed SSL certificates, the format in which you receive these certificates, etc. **Good luck!**

Finally: If you choose to have your Dataverse support secure **Shibboleth authentication**, it will require a server and port configuration that is different still. Under this arrangement Glassfish instance is running on a high local port unaccessible from the outside, and is "hidden" behind Apache. With the latter running on the default https port, accepting and proxying the incoming connections to the former. This is described in the Shibboleth section of the Installation Guide (please note that, at the moment, this functionality is offered as "experimental"). With this proxying setup in place, the TwoRavens and rApache configuration actually becomes simpler. As both the Dataverse and TwoRavens will be served on the same port - 443 (the default port for https). So when running the installer script above, and providing you are planning to run both on the same server, enter "https", your host name and "443" for the rApache protocol, host and port, respectively. The base URL of the Dataverse app will be simply https://{your host name}/.

2.4.4 Appendix

Explained below are the steps needed to manually install and configure the required R packages, and to configure TwoRavens to run under rApache (these are performed by the `r-setup.sh` and `install.pl` scripts above). Provided for reference.

2.4.5 r-setup.sh script:

TwoRavens requires the following R packages and versions to be installed:

R Package	Version Number
Zelig	5.0.5
Rook	1.1.1
rjson	0.2.13
jsonlite	0.9.16
DescTools	0.99.11

Note that some of these packages have their own dependencies, and additional installations are likely necessary. TwoRavens is not compatible with older versions of these R packages.

2.4.6 install.pl script:

I. Configure the TwoRavens web (Javascript) application.

Edit the file `/var/www/html/dataexplore/app_ddi.js`.

find and edit the following 3 lines:

1. `var production=false;`

   and change it to `true`;

2. `hostname="localhost:8080";`

   so that it points 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!)

   and

3. `var rappURL = "http://0.0.0.0:8000/custom/";`

   set this to the URL of your rApache server, i.e.

   `"https://<rapacheserver>:<rapacheport>/custom/";`

II. Configure the R applications to run under rApache

rApache is a loadable httpd module that provides a link between Apache and R. When you installed the rApache rpm, under 0., it placed the module in the Apache library directory and added a configuration entry to the config file (`/etc/httpd/conf/httpd.conf`).

Now we need to configure rApache to serve several R "mini-apps", from the R sources provided with TwoRavens.

a. Edit the following files:

in `dataexplore/rook`:

`rookdata.R, rookzelig.R, rooksubset.R, rooktransform.R, rookselector.R, rooksource.R`

and replace *every* instance of `production<-FALSE` line with `production<-TRUE`.

(yeah, that's why we provide that installer script...)

b. Edit dataexplore/rook/rooksource.R

and change the following line:

```
setwd("/usr/local/glassfish4/glassfish/domains/domain1/docroot/dataexplore/rook")
```

to

```
setwd("/var/www/html/dataexplore/rook")
```

(or your dataexplore directory, if different from the above)

c. Edit the following lines in dataexplore/rook/rookutils.R:

```
url <- paste("https://demo.dataverse.org/custom/preprocess_dir/preprocessSubset_",sessionic
```

and

```
imageVector[[qicount]]<<-paste("https://dataverse-demo.iq.harvard.edu/custom/pic_dir/",
mysessionid,"_",mymodelcount,qicount,".png", sep = "")
```

and change the URL to reflect the correct location of your rApache instance - make sure that the protocol and the port number are correct too, not just the host name!

d. Add the following lines to /etc/httpd/conf/httpd.conf:

(This configuration is now supplied in its own config file `tworavens-rapache.conf`, it can be dropped into the Apache's `/etc/httpd/conf.d`. Again, the scripted installer will do this for you automatically.)

```
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>
```

e. Create the following directories and chown them user apache:

```
mkdir --parents /var/www/html/custom/pic_dir

mkdir --parents /var/www/html/custom/preprocess_dir

mkdir --parents /var/www/html/custom/log_dir

chown -R apache.apache /var/www/html/custom
```

f. chown the dataexplore directory

to user apache:

```
chown -R apache /var/www/html/dataexplore
```

g. restart httpd

```
service httpd restart
```

2.5 Shibboleth

2.5.1 Status: Experimental

Shibboleth support in Dataverse should be considered **experimental** until the following issue is closed: https://github.com/IQSS/dataverse/issues/2117

In Dataverse 4.0, Shibboleth support requires fronting Glassfish with Apache as described below, but this configuration has not been heavily tested in a production environment and is not recommended until this issue is closed: https://github.com/IQSS/dataverse/issues/2180

2.5.2 System Requirements

Only Red Hat Enterprise Linux (RHEL) 6 and derivatives such as CentOS have been tested and only on x86_64. RHEL 7 and Centos 7 **should** work but you'll need to adjust the yum repo config accordingly.

Debian and Ubuntu are not recommended until this issue is closed: https://github.com/IQSS/dataverse/issues/1059

2.5.3 Installation

Install Apache

We include `mod_ssl` for HTTPS support.

```
yum install httpd mod_ssl
```

Install Shibboleth

Enable Shibboleth Yum Repo

This yum repo is recommended at https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPLinuxRPMInstall

```
cd /etc/yum.repos.d
wget http://download.opensuse.org/repositories/security:/shibboleth/CentOS_CentOS-6/securit
```

Install Shibboleth Via Yum

```
yum install shibboleth shibboleth-embedded-ds
```

2.5.4 Configuration

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, 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 when Glassfish 4.2 ships the Dataverse team plans to evaulate if the version of Grizzly included is new enough to include the bug fix, obviating the need for the patch.

Glassfish HTTP and HTTPS ports

Ensure that the Glassfish HTTP service is listening on the default port (8080):

```
asadmin set server-config.network-config.network-listeners.network-listener.http-listener-1
```

Ensure that the Glassfish HTTPS service is listening on the default port (8181):

```
asadmin set server-config.network-config.network-listeners.network-listener.http-listener-2
```

AJP

A `jk-connector` network listener should have already been set up at installation time and you can verify this with `asadmin list-network-listeners` but for reference, here is the command that is used:

```
asadmin create-network-listener --protocol http-listener-1 --listenerport 8009
--jkenabled true jk-connector
```

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.RequestUti` 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

Configure Apache

Enforce HTTPS

To prevent attacks such as FireSheep, HTTPS should be enforced. https://wiki.apache.org/httpd/RewriteHTTPToHTTPS provides a good method. Here is how it looks in a sample file at `/etc/httpd/conf.d/shibtest.dataverse.org.conf`:

```
<VirtualHost *:80>

ServerName shibtest.dataverse.org

# 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 Config Files

`/etc/httpd/conf.d/ssl.conf` should contain the FQDN of your hostname like this: `ServerName shibtest.dataverse.org:443`

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.

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.

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:2.0:native:sp:config" xmlns:md="urn:oasis:names:tc:SAML:2.0:meta
    clockSkew="1800">

    <!-- FIXME: change the entityID to your hostname. -->
    <ApplicationDefaults entityID="https://shibtest.dataverse.org/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

            <SSO>
              SAML2 SAML1
            </SSO>

            <!-- SAML and local-only logout. -->
            <!-- https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPServiceLogout -->
            <Logout>SAML2 Local</Logout>

            <!--
                Handlers allow you to interact with the SP and gather more information.  Try them out
                Attribute values received by the SP through SAML will be visible at:
                http://shibtest.dataverse.org/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. -->
            <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 communi
        <!-- IdPs we want allow go in /etc/shibboleth/dataverse-idp-metadata.xml -->
        <MetadataProvider type="XML" file="dataverse-idp-metadata.xml" backingFilePath="local-idp-met

        <!-- 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>
```

attribute-map.xml

By default, some attributes `/etc/shibboleth/attribute-map.xml` are commented out. Edit the file to enable them.

You can download a sample attribute-map.xml file.

dataverse-idp-metadata.xml

The configuration above 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 .

Disable SELinux

You must set `SELINUX=permisive` in `/etc/selinux/config` and run `setenforce permissive` or otherwise disable SELinux for Shibboleth to work. "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." – https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPSELinux

Restart Apache and Shibboleth

After configuration is complete:

```
service shibd restart
```

```
service httpd restart
```

As a sanity check, visit the following URLs for your hostname to make sure you see JSON and XML:

- https://shibtest.dataverse.org/Shibboleth.sso/DiscoFeed
- https://shibtest.dataverse.org/Shibboleth.sso/Metadata

The JSON in `DiscoFeed` comes from the list of IdPs in `/etc/shibboleth/dataverse-idp-metadata.xml` and will form a dropdown list on the Login Page.

Enable Shibboleth

```
curl -X PUT -d true http://localhost:8080/api/admin/settings/:ShibEnabled
```

After enabling Shibboleth, assuming the `DiscoFeed` is working per above, you should see a list of institutions to log into. You will not be able to log in via these institutions, however, until you have exchanged metadata with them.

2.5.5 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/Native

For discussion of "eppn" vs. other attributes such as "eduPersonTargetedID" or "NameID", please see https://github.com/IQSS/dataverse/issues/1422

2.5.6 Testing

http://testshib.org is a fantastic resource for testing Shibboleth configurations.

First, download your metadata like this (substituting your hostname in both places):

```
curl https://shibtest.dataverse.org/Shibboleth.sso/Metadata >
shibtest.dataverse.org
```

Then upload it to http://testshib.org/register.html

Then try to log in.

Please note that when you try logging in via the TestShib IdP, it is expected that you'll see errors about the "mail" attribute being null. (TestShib is aware of this but it isn't a problem for testing.)

When you are done testing, you can delete the TestShib users you created like this:

```
curl -X DELETE http://localhost:8080/api/admin/authenticatedUsers/myself
```

2.5.7 Backups

It's important to back up these files:

- `/etc/shibboleth/sp-cert.pem`
- `/etc/shibboleth/sp-key.pem`

2.5.8 Feedback

Given that Shibboleth support is experimental and new, feedback is very welcome at support@dataverse.org or via http://community.dataverse.org/community-groups/auth.html .

2.6 Administration

2.6.1 User Administration

Deleting an API token

If an API token is compromised it should be deleted. Users can generate a new one for themselves, but someone with access to the database can delete tokens as well.

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.

After the API token has been deleted, users can generate a new one per Account Creation & Management.

CHAPTER 3

API Guide

We encourage anyone interested in building tools to interoperate with the Dataverse to utilize our APIs. In 4.0, we require to get a token, by simply registering for a Dataverse account, before using our APIs (We are considering making some of the APIs completely public in the future - no token required - if you use it only a few times).

Rather than using a production installation of Dataverse, API users should use http://apitest.dataverse.org for testing.

Contents:

3.1 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, Java, R, Ruby, and PHP.

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 will 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. For a quick introduction to SWORD, the two minute video at http://cottagelabs.com/news/intro-to-sword-2 is recommended.

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.

Contents

3.1.1 Backward incompatible changes

For better security, usernames and passwords are no longer accepted. The use of an API token is required.

In addition, differences in Dataverse 4.0 have 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.

3.1.2 New features as of v1.1

- Dataverse 4.0 supports API tokens and they must be used rather that 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:`.
- 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.

- "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 owners 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.

3.1.3 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.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/service-document
```

Create a dataset with an Atom entry

```
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/
```

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:de
   <!-- 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
   <!-- 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.</d
   <!-- related publications -->
   <dcterms:isReferencedBy holdingsURI="http://dx.doi.org/10.1038/dvn333" agency="DOI" IDNo="10.1038/
</entry>
```

Dublin Core Terms (DC Terms) Qualified Mapping - Dataverse DB Element Crosswalk

DC (terms: namespace)	Dataverse DB Element	Required	Note
dcterms:title	title	Y	Title of the Dataset.
dcterms:creator	authorName (LastName, FirstName)	Y	Author(s) for the Dataset.
dcterms:subject	subject (Controlled Vocabulary) OR keyword	Y	Controlled Vocabulary list is in our User Guide > Metadata References.
dcterms:description	dsDescriptionValue	Y	Describing the purpose, scope or nature of the Dataset. Can also use dcterms:abstract.
dcterms:publisher	producerName		Person or agency financially or administratively responsible for the Dataset
dcterms: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.
dcterms:date	productionDate (YYYY-MM-DD or YYYY-MM or YYYY)		Production date of Dataset.
dcterms:type	kindOfData		Type of data included in the file: survey data, census/enumeration data, aggregate data, clinical.
dcterms:source	dataSources		List of books, articles, data files if any that served as the sources for the Dataset.
dcterms:relation	relatedMaterial		Any related material (journal article citation is not included here - see: dcterms:isReferencedBy below).
dcterms:coverage	otherGeographicCoverage		General information on the geographic coverage of the Dataset.
dcterms:license	license		Set the license to CC0 (default in Dataverse for new Datasets), otherwise enter "NONE" and fill in the dcterms:rights field.
dcterms:rights	termsofuse		If not using CC0, enter any terms of use or restrictions for the Dataset.
dcterms: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

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/collection/datavers
```

Add files to a dataset with a zip file

```
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

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
```

Display a dataset statement

Contains title, author, feed of file entries, latestVersionState, locked boolean, updated timestamp.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/statement/study/doi
```

Delete a file by database id

```
curl -u $API_TOKEN: -X DELETE https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit-medi
```

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.

```
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/e
```

Delete a dataset

```
curl -u $API_TOKEN: -i -X DELETE https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/edit/s
```

Determine if a dataverse has been published

Look for a *dataverseHasBeenReleased* boolean.

```
curl -u $API_TOKEN: https://$HOSTNAME/dvn/api/data-deposit/v1.1/swordv2/collection/datavers
```

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.

```
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/$DATAVE
```

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.

```
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/12
```

3.1.4 Known issues

- Potential mismatch between the dataverses ("collections" from a SWORD perspective) the user can deposit data into in returned by the Service Document and which dataverses the user can actually deposit data into. This is due to an incomplete transition from the old DVN 3.x "admin-only" style permission checking to the new permissions system in Dataverse 4.0 (https://github.com/IQSS/dataverse/issues/1070). The mismatch was reported at https://github.com/IQSS/dataverse/issues/1443
- Should see all the fields filled in for a dataset regardless of what the parent dataverse specifies: https://github.com/IQSS/dataverse/issues/756
- Inefficiency in constructing the `Service Document`: https://github.com/IQSS/dataverse/issues/784
- Inefficiency in constructing the list of datasets: https://github.com/IQSS/dataverse/issues/784

3.1.5 Roadmap

These are features we'd like to add in the future:

- Implement SWORD 2.0 Profile 6.4: https://github.com/IQSS/dataverse/issues/183
- Support deaccessioning via API: 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 dcterms xml element for equivalent of required field not found: https://github.com/IQSS/dataverse/issues/1019

3.1.6 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

3.1.7 Client libraries

- Python: https://github.com/swordapp/python-client-sword2
- Java: https://github.com/swordapp/JavaClient2.0
- R: https://github.com/ropensci/dvn
- Ruby: https://github.com/swordapp/sword2ruby
- PHP: https://github.com/swordapp/swordappv2-php-library

3.2 Search API

3.2.1 About

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 until issue 1299 is resolved.

The parameters and JSON response are partly inspired by the GitHub Search API.

3.2.2 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, https://apitest.dataverse.org/api/search?q=title:data
type	string	Can be either "dataverse", "dataset", or "file". Multiple "type" parameters can be used to include multiple types (i.e. `type=dataset&type=file`). If omitted, all types will be returned. For example, https://apitest.dataverse.org/api/search?q=*&type=dataset
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. For example, https://apitest.dataverse.org/api/search?q=data&subtree=birds
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, https://apitest.dataverse.org/api/search?q=data&sort=name&order=asc
per_page	int	The number of results to return per request. The default is 10. The max is 1000. See *iteration example*.
start	int	A cursor for paging through search results. See *iteration example*.
show_relevance	boolean	Whether or not to show details of which fields were matched by the query. False by default. See *advanced search example*.
show_facets	boolean	Whether or not to show facets that can be operated on by the "fq" parameter. False by default. See *advanced search example*.
fq	string	A filter query on the search term. Multiple "fq" parameters can be used. See *advanced search example*.

3.2.3 Basic Search Example

https://apitest.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://apitest.dataverse.org/dataverse/trees",
                "image_url":"https://apitest.dataverse.org/api/access/dvCardImage/7",
                "identifier":"trees",
                "description":"A tree dataverse with some birds",
                "published_at":"2015-01-12T16:05:12Z"
            },
            {
                "name":"Chestnut Trees",
                "type":"dataverse",
                "url":"https://apitest.dataverse.org/dataverse/chestnuttrees",
                "image_url":"https://apitest.dataverse.org/api/access/dvCardImage/9",
                "identifier":"chestnuttrees",
                "description":"A dataverse with chestnut trees and an oriole",
                "published_at":"2015-01-12T18:02:32Z"
            },
            {
                "name":"trees.png",
                "type":"file",
                "url":"https://apitest.dataverse.org/api/access/datafile/12",
                "image_url":"https://apitest.dataverse.org/api/access/preview/12",
                "file_id":"12",
                "description":"",
                "published_at":"2015-01-12T16:05:44Z",
                "file_type":"PNG Image",
                "size_in_bytes":8361,
                "md5":"0386269a5acb2c57b4eade587ff4db64",
                "dataset_citation":"Spruce, Sabrina, 2015, \"Spruce Goose\", http://dx.doi.org/10.50
            },
            {
                "name":"Birds",
                "type":"dataverse",
                "url":"https://apitest.dataverse.org/dataverse/birds",
                "image_url":"https://apitest.dataverse.org/api/access/dvCardImage/2",
                "identifier":"birds",
                "description":"A bird dataverse with some trees",
                "published_at":"2015-01-12T18:01:51Z"
            }
        ],
        "count_in_response":4
    }
}
```


3.2.4 Advanced Search Example

https://apitest.dataverse.org/api/search?q=finch&show_relevance=true&show_facets=true&fq=publication_date_s:2015&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=publication_date_s:2015`. 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://apitest.dataverse.org/dataverse/finches",
                "image_url":"https://apitest.dataverse.org/api/access/dvCardImage/3",
                "identifier":"finches",
                "description":"A dataverse with finches",
                "published_at":"2015-01-12T18:01:15Z",
                "matches":[
                    {
                        "description":{
                            "snippets":[
                                "A dataverse with <span class=\"search-term-match\">finches</span>"
                            ]
                        }
                    },
                    {
                        "name":{
                            "snippets":[
                                "<span class=\"search-term-match\">Finches</span>"
                            ]
                        }
                    }
                ]
            },
            {
                "name":"Darwin's Finches",
                "type":"dataset",
                "url":"http://dx.doi.org/10.5072/FK2/CE0052",
                "image_url":"https://apitest.dataverse.org/api/access/dsPreview/2",
                "global_id":"doi:10.5072/FK2/CE0052",
                "published_at":"2015-01-12T18:01:37Z",
                "citation":"Finch, Fiona, 2015, \"Darwin's Finches\", http://dx.doi.org/10.5072/FK2/C
                "description": "Darwin's finches (also known as the Galápagos finches) are a group of
                "matches":[
                    {
                        "authorName":{
                            "snippets":[
                                "<span class=\"search-term-match\">Finch</span>, Fiona"
                            ]
                        }
                    },
                    {
                        "dsDescriptionValue":{
                            "snippets":[
                                "Darwin's <span class=\"search-term-match\">finches</span> (also know
                            ]
                        }
                    },
                    {
                        "title":{
```

```
                            "snippets":[
                                "Darwin's <span class=\"search-term-match\">Finches</span>"
                            ]
                        }
                    }
                ],
                "authors":[
                    "Finch, Fiona"
                ]
            }
        ],
        "facets":[
            {
                "dvCategory_s":{
                    "friendly":"Dataverse Category",
                    "labels":[
                        {
                            "Uncategorized":1
                        }
                    ]
                },
                "affiliation_ss":{
                    "friendly":"Affiliation",
                    "labels":[
                        {
                            "Birds Inc.":1
                        }
                    ]
                },
                "publication_date_s":{
                    "friendly":"Publication Date",
                    "labels":[
                        {
                            "2015":2
                        }
                    ]
                }
            }
        ],
        "count_in_response":2
    }
}
```


3.2.5 Iteration

Be 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://apitest.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)
```

3.3 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.

3.3.1 Basic File Access

Basic acces URI:

`/api/access/datafile/$id`

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. (TODO: *get a proper description of the feature from James/Vito*)

`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.

`vars`

For column-wise subsetting (available for tabular data files only).

Example:

`http://localhost:8080/api/meta/datafile/6?vars=123,127`

where 123 and 127 are the ids of data variables that belong to the data file with the id 6.

3.3.2 Multiple File ("bundle") download

`/api/access/datafiles/$id1,$id2,...$idN`

Returns the files listed, zipped.

Parameters:

none.

3.3.3 "All Formats" bundled access for Tabular Files.

`/api/access/datafile/bundle/$id`

This is convenience packaging method is 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:

none.

3.3.4 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 XML is the only format supported so far. In the future, support for formatting the output in JSON will (may?) be added.

The DDI returned will only have 2 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/meta/datafile/6`

```
<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+6
   </fileDscr>
   <dataDscr>
      <var ID="v1" name="id" intrvl="discrete">
         <location fileid="f6"/>
         <labl level="variable">Personen-ID</labl>
         <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:AvELPR5
      </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:XqQaMw(
      </var>
   </dataDscr>
</codeBook>
```

More information on the DDI is available at (TODO).

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.

Partial record parameters:

(TODO).

`/api/access/datafile/$id/metadata/preprocessed`

This method provides the "Pre-processed 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.

3.3.5 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=...`.

3.4 Native API

Dataverse 4.0 exposes most of its GUI functionality via a REST-based API. Some API calls do not require authentication. Calls that do require authentication require the user's API key. That key can be passed either via an extra query parameter, `key`, as in `ENPOINT?key=API_KEY`, or via the HTTP header `X-Dataverse-key`. Note that while the header option normally requires more work on client side, it is considered safer, as the API key is not logged in the server access logs.

Warning: Dataverse 4.0'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

3.4.1 Endpoints

Dataverses

Generates a new dataverse under `$id`. Expects a json content describing the dataverse. If `$id` is omitted, a root dataverse is created. `$id` can either be a dataverse id (long) or a dataverse alias (more robust).

```
POST http://$SERVER/api/dataverses/$id?key=$apiKey
```

View data about the dataverse identified by `$id`. `$id` can be the id number of the dataverse, its alias, or the special value `:root`.

```
GET http://$SERVER/api/dataverses/$id
```

Deletes the dataverse whose ID is given:

```
DELETE http://$SERVER/api/dataverses/$id?key=$apiKey
```

Lists all the DvObjects under dataverse `id`.

```
GET http://$SERVER/api/dataverses/$id/contents
```

All the roles defined directly in the dataverse identified by `id`:

```
GET http://$SERVER/api/dataverses/$id/roles?key=$apiKey
```

List all the facets for a given dataverse `id`.

```
GET http://$SERVER/api/dataverses/$id/facets?key=$apiKey
```

Creates a new role under dataverse `id`. Needs a json file with the role description:

```
POST http://$SERVER/api/dataverses/$id/roles?key=$apiKey
```

List all the role assignments at the given dataverse:

```
GET http://$SERVER/api/dataverses/$id/assignments?key=$apiKey
```

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 the assignment whose id is `$id`:

```
DELETE http://$SERVER/api/dataverses/$id/assignments/$id?key=$apiKey
```

Get the metadata blocks defined on the passed dataverse:

```
GET http://$SERVER/api/dataverses/$id/metadatablocks?key=$apiKey
```

Sets the metadata blocks of the dataverse. Makes the dataverse a metadatablock root. The query body is a JSON array with a list of metadatablocks identifiers (either id or name).

```
POST http://$SERVER/api/dataverses/$id/metadatablocks?key=$apiKey
```

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
```

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).

```
POST http://$SERVER/api/dataverses/$id/metadatablocks/:isRoot?key=$apiKey
```

Create a new dataset in dataverse `id`. The post data is a Json object, containing the dataset fields and an initial dataset version, under the field of `"datasetVersion"`. The initial versions version number will be set to `1.0`, and its state will be set to `DRAFT` regardless of the content of the json object. Example json can be found at `data/dataset-create-new.json`.

```
POST http://$SERVER/api/dataverses/$id/datasets/?key=$apiKey
```

Publish the Dataverse pointed by `identifier`, which can either by the dataverse alias or its numerical id.

```
POST http://$SERVER/api/dataverses/$identifier/actions/:publish?key=$apiKey
```

Datasets

Note Creation of new datasets is done by ‘‘POST‘‘ing them onto dataverses. See dataverse 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`

Show the dataset whose id is passed:

```
GET http://$SERVER/api/datasets/$id?key=$apiKey
```

Delete the dataset whose id is passed:

```
DELETE http://$SERVER/api/datasets/$id?key=$apiKey
```

List versions of the dataset:

```
GET http://$SERVER/api/datasets/$id/versions?key=$apiKey
```

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
```

Lists all the file metadata, for the given dataset and version:

```
GET http://$SERVER/api/datasets/$id/versions/$versionId/files?key=$apiKey
```

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
```

Lists the metadata block block named *blockname*, for the given dataset and version:

```
GET http://$SERVER/api/datasets/$id/versions/$versionId/metadata/$blockname?key=$apiKey
```

Updates the current draft version of dataset `$id`. If the dataset does not have an draft version - e.g. when its most recent version is published, a new draft version is created. The invariant is - after a successful call to this command, the dataset has a DRAFT version with the passed data. The request body is a dataset version, in json format.

```
PUT http://$SERVER/api/datasets/$id/versions/:draft?key=$apiKey
```

Publishes the dataset whose id is passed. 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):

```
POST http://$SERVER/api/datasets/$id/actions/:publish?type=$type&key=$apiKey
```

Deletes the draft version of dataset `$id`. Only the draft version can be deleted:

```
DELETE http://$SERVER/api/datasets/$id/versions/:draft?key=$apiKey
```

Builtin Users

This endopint deals with users of the built-in authentication provider. Note that users may come from other authentication services as well, such as Shibboleth. For this service to work, the setting `BuiltinUsers.KEY` has to be set, and its value passed as `key` to each of the calls.

Generates a new user. Data about the user are posted via JSON. *Note that the password is passed as a parameter in the query*.

```
POST http://$SERVER/api/builtin-users?password=$password&key=$key
```

Gets the API token of the user, given the password.

```
GET http://$SERVER/api/builtin-users/$username/api-token?password=$password
```

Roles

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
```

Shows the role with `id`:

```
GET http://$SERVER/api/roles/$id
```

Deletes the role with `id`:

```
DELETE http://$SERVER/api/roles/$id
```

Explicit Groups

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 under dataverse `$id`:

```
GET http://$server/api/dataverses/$id/groups
```

Show group `$groupAlias` under dataverse `$dv`:

```
GET http://$server/api/dataverses/$dv/groups/$groupAlias
```

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 `$groupAlias` under dataverse `$dv`:

```
DELETE http://$server/api/dataverses/$dv/groups/$groupAlias
```

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 single role assignee to a group. Request body is ignored:

```
PUT http://$server/api/dataverses/$dv/groups/$groupAlias/roleAssignees/$roleAssigneeIdentifier
```

Remove a single role assignee from an explicit group:

```
DELETE http://$server/api/dataverses/$dv/groups/$groupAlias/roleAssignees/$roleAssigneeIdentifier
```

Metadata Blocks

Lists brief info about all metadata blocks registered in the system:

```
GET http://$SERVER/api/metadatablocks
```

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
```

Admin

This is the administrative part of the API. It is probably a good idea to 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.

List all settings:

```
GET http://$SERVER/api/admin/settings
```

Sets setting `name` to the body of the request:

```
PUT http://$SERVER/api/admin/settings/$name
```

Get the setting under `name`:

```
GET http://$SERVER/api/admin/settings/$name
```

Delete the setting under `name`:

```
DELETE http://$SERVER/api/admin/settings/$name
```

List the authentication provider factories. The alias field of these is used while configuring the providers themselves.

```
GET http://$SERVER/api/admin/authenticationProviderFactories
```

List all the authentication providers in the system (both enabled and disabled):

```
GET http://$SERVER/api/admin/authenticationProviders
```

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 data about an authentication provider:

```
GET http://$SERVER/api/admin/authenticationProviders/$id
```

Enable or disable an authentication provider (denoted by `id`):

```
POST 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/authentic
```

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 all global roles in the system.

```
GET http://$SERVER/api/admin/roles
```

Creates a global role in the Dataverse installation. The data POSTed are assumed to be a role JSON.

```
POST http://$SERVER/api/admin/roles
```

Toggles superuser mode on the `AuthenticatedUser` whose `identifier` is passed.

```
POST http://$SERVER/api/admin/superuser/$identifier
```

IpGroups

List all the ip groups:

```
GET http://$SERVER/api/admin/groups/ip
```

Adds a new ip group. POST data should specify the group in JSON format. Examples are available at `data/ipGroup1.json`.

```
POST http://$SERVER/api/admin/groups/ip
```

Returns a the group in a JSON format. `groupIdtf` can either be the group id in the database (in case it is numeric), or the group alias.

```
GET http://$SERVER/api/admin/groups/ip/$groupIdtf
```

Deletes the group specified by `groupIdtf`. `groupIdtf` can either be the group id in the database (in case it is numeric), or the group alias. Note that a group can be deleted only if there are no roles assigned to it.

```
DELETE http://$SERVER/api/admin/groups/ip/$groupIdtf
```

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
```

3.5 Client Libraries

Currently there are client libraries for Python and R that can be used to develop against Dataverse APIs.

Because Dataverse is a SWORD server, additional client libraries exist for Java, Ruby, and PHP per the SWORD API page.

3.5.1 Python

https://github.com/IQSS/dataverse-client-python is the offical Python package for Dataverse APIs.

Robert Liebowitz from the Center for Open Science heads its development and the library is used to integrate the Open Science Framework (OSF) with Dataverse via an add-on which itself is open source and listed on the Apps page.

3.5.2 R

https://github.com/IQSS/dataverse-client-r is the official R package for Dataverse APIs.

It was created by Thomas Leeper whose dataverse can be found at https://dataverse.harvard.edu/dataverse/leeper

3.6 Apps

The introduction of Dataverse APIs has fostered the development of apps that are listed at http://datascience.iq.harvard.edu/collaborations

The apps below are open source, demonstrating how to use Dataverse APIs. Some of these apps (and others) are built on Client Libraries that are available for Dataverse APIs.

- *Javascript*
  - *TwoRavens*
- *PHP*
  - *OJS*
- *Python*
  - *OSF*
  - *GeoConnect*
- *Java*
  - *Dataverse for Android*

3.6.1 Javascript

TwoRavens

TwoRavens is a system of interlocking statistical tools for data exploration, analysis, and meta-analysis.

https://github.com/IQSS/TwoRavens

3.6.2 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_6/plugins/generic/dataverse

3.6.3 Python

OSF

Allows you to view, download, and upload files to and from a Dataverse dataset from an Open Science Framework (OSF) project: https://osf.io/getting-started/#dataverse

https://github.com/CenterForOpenScience/osf.io/tree/master/website/addons/dataverse

GeoConnect

GeoConnect allows Dataverse files to be visualized on http://worldmap.harvard.edu with a "Map It" button.

https://github.com/IQSS/geoconnect

3.6.4 Java

Dataverse for Android

For now this is only a proof of concept.

https://github.com/IQSS/dataverse-android

CHAPTER 4

Developers' Guide

Contents:

4.1 Introduction

Welcome! Dataverse is an open source project that loves contributors!

4.1.1 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

To get started, you'll want to set up your Development Environment and make sure you understand the Branching Strategy. Thorough Testing is encouraged (and expected). Opinions about Coding Style are welcome!

If you have any questions at all, please reach out to other developers per https://github.com/IQSS/dataverse/blob/master/CONTRIBUTING.md

4.1.2 Roadmap

For the Dataverse development roadmap, please see https://github.com/IQSS/dataverse/milestones

The Contributing to Dataverse document in the root of the source tree provides guidance on:

- the use of labels to organize and prioritize issues
- making pull requests
- how to contact the development team

4.1.3 Related Guides

If you are a developer who wants to make use of Dataverse APIs, please see the API 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.

4.1.4 Related Projects

As a developer, you also may be interested in these projects related to Dataverse:

- Zelig (R) - run statistical models on files uploaded to Dataverse: https://github.com/IQSS/Zelig
- TwoRavens (Javascript) - a d3.js interface for exploring data and running Zelig models: https://github.com/IQSS/TwoRavens
- Universal Numerical Fingerprint (UNF) (Java) - a Universal Numerical Fingerprint: https://github.com/IQSS/UNF
- DataTags (Java and Scala) - tag datasets with privacy levels: https://github.com/IQSS/DataTags
- GeoConnect (Python) - create a map by uploading files to Dataverse: https://github.com/IQSS/geoconnect
- Dataverse API client libraries - use Dataverse APIs from various languages: Client Libraries
- Third party apps - make use of Dataverse APIs: Apps
- [Your project here] :)

4.2 Development Environment

4.2.1 Assumptions

This guide assumes you are using a Mac but we do have pages for `/developers/windows` and `/developers/ubuntu`.

4.2.2 Requirements

Java

Dataverse is developed on Java 7. An upgrade to Java 8 is being tracked at https://github.com/IQSS/dataverse/issues/2151

The use of Oracle's version of Java is recommended, which can be downloaded from http://www.oracle.com/technetwork/java/javase/downloads/index.html

The version of OpenJDK available from package managers from common Linux distributions such as Ubuntu and Fedora is probably sufficient for small changes as well as day to day development.

Glassfish

As a Java Enterprise Edition 7 (Java EE 7) application, Dataverse requires an applications server to run.

Glassfish 4.1+ is required, which can be downloaded from http://glassfish.java.net

PostgreSQL

PostgreSQL 9.x is required and can be downloaded from http://postgresql.org

Solr

Dataverse depends on Solr for browsing and search.

Solr 4.6.0 is the only version that has been tested extensively and is recommended in development. Download and configuration instructions can be found below. An upgrade to newer versions of Solr is being tracked at https://github.com/IQSS/dataverse/issues/456

curl

A command-line tool called `curl` (http://curl.haxx.se) is required by the setup scripts and it is useful to have curl installed when working on APIs.

jq

A command-line tool called `jq` (http://stedolan.github.io/jq/) is required by the setup scripts.

If you are already using `brew`, `apt-get`, or `yum`, you can install `jq` that way. Otherwise, download the binary for your platform from http://stedolan.github.io/jq/ and make sure it is in your `$PATH` (`/usr/bin/jq` is fine) and executable with `sudo chmod +x /usr/bin/jq`.

4.2.3 Recommendations

Mac OS X

The setup of a Dataverse development environment assumes the presence of a Unix shell (i.e. bash) so an operating system with Unix underpinnings such as Mac OS X or Linux is recommended. (The development team at IQSS has standardized Mac OS X.) Windows users are encouraged to install Cygwin.

Netbeans

While developers are welcome to use any editor or IDE they wish, Netbeans 8+ is recommended because it is free of cost, works cross platform, has good support for Java EE projects, and happens to be the IDE that the development team at IQSS has standardized on.

NetBeans can be downloaded from http://netbeans.org. Please make sure that you use an option that contains the Jave EE features when choosing your download bundle. While using the installer you might be prompted about installing JUnit and Glassfish. There is no need to reinstall Glassfish, but it is recommended that you install JUnit.

This guide will assume you are using Netbeans for development.

Additional Tools

Please see also the Tools page, which lists additional tools that very useful but not essential.

4.2.4 Setting up your dev environment

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.

Clone Project from GitHub

Before making commits, please read about our Branching Strategy to make sure you commit to the right branch.

Determine Which Repo To Push To

Developers who are not part of the development team at IQSS should first fork https://github.com/IQSS/dataverse per https://help.github.com/articles/fork-a-repo/

Cloning the Project from Netbeans

From NetBeans, click "Team" then "Remote" then "Clone". Under "Repository URL", enter the "ssh clone URL" for your fork (if you do not have push access to the repo under IQSS) or `git@github.com:IQSS/dataverse.git` (if you do have push access to the repo under IQSS). See also https://netbeans.org/kb/docs/ide/git.html#github

Cloning the Project from the Terminal

If you prefer using git from the command line, you can clone the project from a terminal and later open the project in Netbeans.

If you do not have push access to https://github.com/IQSS/dataverse clone your fork:

```
git clone git@github.com:[your GitHub user or organization]/dataverse.git
```

If you do have push access to https://github.com/IQSS/dataverse clone it:

```
git clone git@github.com:IQSS/dataverse.git
```

Installing and Running Solr

A Dataverse-specific `schema.xml` configuration file (described below) is required.

Download solr-4.6.0.tgz from http://archive.apache.org/dist/lucene/solr/4.6.0/solr-4.6.0.tgz to any directory you like but in the example below, we have downloaded the tarball to a directory called "solr" in our home directory. For now we are using the "example" template but we are replacing `schema.xml` with our own. We will also assume that the clone on the Dataverse repository was retrieved using NetBeans and that it is saved in the path ~/NetBeansProjects.

- `cd ~/solr`
- `tar xvfz solr-4.6.0.tgz`
- `cd solr-4.6.0/example`
- `cp ~/NetBeansProjects/dataverse/conf/solr/4.6.0/schema.xml solr/collection1/conf/schema.xml`
- `java -jar start.jar`

Please note: If you prefer, once the proper `schema.xml` file is in place, you can simply double-click "start.jar" rather that 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.

Once Solr is up and running you should be able to see a "Solr Admin" dashboard at http://localhost:8983/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 . You can also get JSON output of static fields Solr knows about: http://localhost:8983/solr/schema/fields

Run installer

Once you install Glassfish and PostgreSQL, you need to configure the environment for the Dataverse app - configure the database connection, set some options, etc. We have a new installer script that should do it all for you. Again, assuming that the clone on the Dataverse repository was retrieved using NetBeans and that it is saved in the path ~/NetBeansProjects:

```
cd ~/NetBeansProjects/dataverse/scripts/installer
```

```
./install
```

The script will prompt you for some configuration values. It is recommended that you choose "localhost" for your hostname if this is a development environment. For everything else it should be safe to accept the defaults.

The script is a variation of the old installer from DVN 3.x that calls another script that runs `asadmin` commands. A serious advantage of this approach is that you should now be able to safely run the installer on an already configured system.

All the future changes to the configuration that are Glassfish-specific and can be done through `asadmin` should now go into `scripts/install/glassfish-setup.sh`.

4.2.5 Shibboleth

If you are working on anything related to users, please keep in mind that your changes will likely affect Shibboleth users. 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.

4.2.6 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 per the "Installing and Running Solr" section of this guide
- 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

4.3 Branching Strategy

4.3.1 Goals

The goals of the Dataverse branching strategy are twofold:

- have developers "stay together" in the same release branch (i.e. 4.0.1), resolving conflicts early
- allow for concurrent development in multiple branches, for example:
  - hot fix branch created from 4.0 tag (i.e. 4.0-patch.1)
  - bug fixes in an unreleased 4.0.1 release branch found in QA
  - feature development in an upcoming 4.0.2 release branch

Release branches that match milestones and future version numbers (i.e. 4.0.1, 4.0.2) are used to achieve these goals. Think of release branches as trains. If you miss the 4.0.1 train, hopefully you'll catch the 4.0.2! The goal is always to get the best code into each release.

4.3.2 Persistent Branches

The "master" branch is the only persistent branch. Commits should never be made directly to master. Commits are only made to release branches. Release branches are merged in "master" just before tagging per Making Releases.

4.3.3 Release Branches

When developers feel the code in a release branch (i.e. 4.0.1) is ready, it is passed to QA. At this point feature development in that release branch (i.e. 4.0.1) is frozen and a new release branch (i.e. 4.0.2) is created from that commit. The frozen release branch is sent to QA for testing.

4.3.4 Feature Development

Developers who have push access to https://github.com/IQSS/dataverse are welcome to push commits directly to the branch corresponding to the milestone the issue tracking the feature has been assigned. For example, if you are working on https://github.com/IQSS/dataverse/issues/2088 and the issue has been assigned to milestone 4.0.1, you are welcome to push directly to the 4.0.1 branch.

Developers who do not have push access should first read https://github.com/IQSS/dataverse/blob/master/CONTRIBUTING.md for general guidelines about contributing code and contacting developers who have the ability to (hopefully!) merge in your contribution. You will likely be advised to submit a pull request against the current release branch that is not yet frozen. For example, if 4.0.1 is frozen (i.e. in QA) the pull request should be made against 4.0.2.

4.3.5 Fixing Bugs in Unreleased Code

Bugs in the release branch currently in QA (i.e. 4.0.1) will be fixed in that branch referencing an issue number. Assuming the fix is good and passes QA, the release branch in QA (i.e. 4.0.1) will be merged into the release branch currently under feature development (i.e. 4.0.2) to get the bug fix.

4.3.6 Fixing Bugs in Released code

Bugs found in released code will (like features) be assigned to milestones visible at https://github.com/IQSS/dataverse/milestones

4.4 Testing

4.4.1 Unit Tests

Write them. JUnit is your friend.

4.4.2 Integration Tests

Write them. REST-assured and Selenium are recommended. A Jenkins environment is available for automated testing: https://build.hmdc.harvard.edu:8443

4.5 Documentation

4.5.1 Quick Fix

If you find a typo or a small error in the documentation you can easily fix it using GitHub.

- Fork the repository
- Go to [your GitHub username]/dataverse/doc/sphinx-guides/source and access the file you would like to fix
- Switch to the branch that is currently under development
- Click the Edit button in the upper-right corner and fix the error
- Submit a pull request

4.5.2 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 the 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 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/user/index.html). Now, using your preferred text editor, open and edit these files, or create new .rst files and edit the others accordingly.

Once you are done, open a terminal and change directories to ~/dataverse/doc/sphinx-guides . Then, run the following commands:

- `make clean`
- `make html Makefile`

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 dataverse repository.

4.6 Debugging

4.6.1 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.

4.7 Coding Style

Like all development teams, the Dataverse developers at IQSS have their habits and styles when it comes to writing code.

A lot of it isn't written down, but a draft has been started at https://docs.google.com/document/d/1KTd3FpM1BI3HlBofaZjMmBiQEJtFf

Debate and comments are welcome!

4.8 Making Releases

4.8.1 Bump Version Numbers

Before tagging, ensure the version number has been incremented in the following places:

- pom.xml (and scripts that reference the name of the war file)
- src/main/java/VersionNumber.properties
- doc/sphinx-guides/source/conf.py

4.8.2 Finalize Documentation

The source for user-facing documentation (including this very guide) is under https://github.com/IQSS/dataverse/tree/master/doc

Docs don't write themselves. Please help out! Before a release is tagged documentation related to that release should be checked in to the release branch (i.e. 4.0.1), ultimately to be hosted under a version number at http://guides.dataverse.org

4.8.3 Write Release Notes

See http://keepachangelog.com

4.8.4 Merge Release Branch

The release branch (i.e. 4.0.1) should be merged into "master" before tagging.

4.8.5 Tag the Release

The tag will be the number of the milestone and release branch (i.e. 4.0.1).

4.8.6 Make Release Available for Download

Upload the following to https://github.com/IQSS/dataverse/releases

- installer (for new installs)
- war file
- database migration script

4.9 Tools

These are handy tools for your Development Environment.

4.9.1 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

4.9.2 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

4.9.3 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.

4.9.4 Vagrant

Vagrant allows you to spin up a virtual machine running Dataverse on your development workstation.

From the root of the git repo, run `vagrant up` and eventually you should be able to reach an installation of Dataverse at http://localhost:8888 (or whatever forwarded_port indicates in the Vagrantfile)

The Vagrant environment can also be used for Shibboleth testing in conjunction with PageKite configured like this:

service_on = http:@kitename : localhost:8888 : @kitesecret

service_on = https:@kitename : localhost:9999 : @kitesecret

Please note that before running `vagrant up` for the first time, you'll need to ensure that required software (GlassFish, Solr, etc.) is available within Vagrant. If you type `cd downloads` and `./download.sh` the software should be properly downloaded.

4.9.5 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.
```

4.10 Universal Numerical Fingerprint (UNF)

Contents:

4.10.1 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)

4.10.2 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.

4.10.3 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.

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.fffff`. `hh`, `mm` and `ss` are 2 digit, zero-padded numbers. `fffff` 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: `Do5dfAoOOFt4FSj0JcByEw==`
Printable UNF: `UNF:6:Do5dfAoOOFt4FSj0JcByEw==`

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+l4ywdwsJeDze8dplJA==`. 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.

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

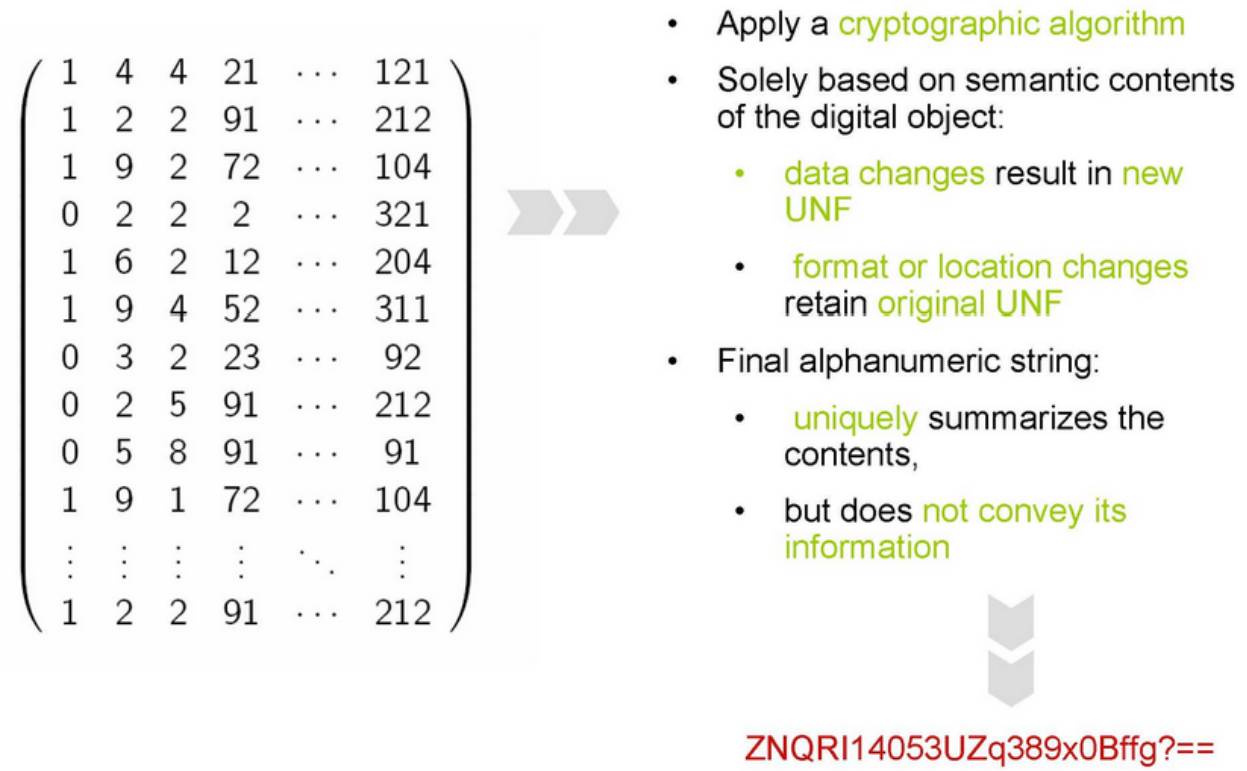


Fig. 4.1: Fig.1 UNF: used to uniquely identify and verify data.

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, Jeff Gill and Michael McDonald, 2003, Numerical Issues in Statistical Computing for the Social Scientist, New York: John Wiley.

CHAPTER 5

How the Guides Are Organized

The Guides are reference documents that explain how to use Dataverse, which are divided into the following sections: User Guide, Installation Guide, Developers Guide, and API 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 Developers 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.

CHAPTER 6

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 shares 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's free of charge. We attempt to respond within one business day to all questions and if it cannot be resolved immediately, we'll let you know what to expect.

The support email address is support@dataverse.org.

This is the same address as the Report Issue link. We try to respond within one business day.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search