Try out DSpace 8

Overview

DSpace 8 brings new features and major updates to the DSpace platform. *See the DSpace 8 Release Notes for more information on this release.*

Try it out using the Sandbox Site

If you want to quickly test out the latest DSpace 8 code, you may do so using our demo site.

- **DSpace 8 User Interface Demo: https://sandbox.dspace.org/** (Login: dspacedemo+admin@gmail.com , Password: dspace)
- **DSpace 8 REST API Demo: https://sandbox.dspace.org/server/** (Login: dspacedemo+admin@gmail.com , Password: dspace)
    - REST Contract / Documentation is available at: https://github.com/DSpace/RestContract/#readme

Keep in mind, our User Interface Demo uses the REST API Demo as the backend. So, any content created (by anyone) will immediately appear in both locations.

Installation

Currently, we offer three different ways to install / run DSpace locally. The route you choose may depend on exactly what you'd like to try out. *Choose ONE of the following:*

1. Install via Docker (*Recommended if you just want to try it out quickly for dev/test environments*) - This is the easiest/quickest way to get *everything* running locally (no matter your technical skills). We even have a script to automatically load some test data and test accounts into your Docker instance.
2. Install the User Interface only, while using the demo REST API backend - This is the easiest way to try out just the User Interface locally. Keep in mind, by default, the User Interface will be installed to use our DSpace REST API Demo as a backend. Therefore, you will be interacting with the content on that demo site and will need to use the login information for the demo site (see above)
3. Install *both* the User Interface and REST API locally **(*Required for Production)***. This is a fully local installation. The UI and REST API need not be installed on the same server.

Install via Docker

Detailed Docker Compose documentation available in the codebase

The below instructions provide a quick way to start up both the User Interface and REST API in a Docker environment. However, if you are already familiar with Docker or plan to use it more heavily, you may wish to also review our detailed Docker Compose instructions for both the frontend (User Interface) and backend (REST API).

Running DSpace via Docker allows you to quickly & easily install and start up all the "parts" of DSpace (database, Solr index, REST API, user interface) without having to install them all yourself. This will work on *any operating system* (Windows 10, Mac OS, or Linux). *Keep in mind, this setup will REQUIRE at least 6GB of memory allocated for Docker. However, 8GB is recommended, if you have it available.*

1. First, install the necessary prerequisites: Docker and Git
    a. On Windows 10:
        i. Install Docker Desktop for Windows
        ii. Install Git Bash & verify the installation by running "`git version`" from Command Prompt or Powershell
    b. On Mac OS:
        i. Install Docker Desktop for Mac
        ii. Git may already be installed. Run "`git version`" from your terminal to see if it's available. If Git is not yet installed, you will be prompted to install developer tools. If that approach doesn't work, Git may also be downloaded.
    c. On Linux:
        i. Install Docker Engine for your Linux OS
        ii. Git may already be installed. Run "`git version`" to check for it. If it is not installed, install Git using your package manager (e.g. `sudo apt-get install git`)
2. Next, using Git, download (clone) the DSpace Angular UI codebase & move into that codebase directory

```
# Download the UI codebase
git clone https://github.com/DSpace/dspace-angular.git

# Move into the created codebase directory
cd dspace-angular

# Switch to the main branch (latest pre-8.0 code)
git checkout main
```

3. Using the Docker instructions in that codebase, start up **both** the DSpace 7 REST API and Angular UI via Docker
    a. First, pull down the latest version of the Docker images (both frontend and backend):

    ```
    docker-compose -f docker/docker-compose.yml -f docker/docker-compose-rest.yml pull
    ```

    b. *Optionally*, you can choose to locally rebuild the Angular UI (This is only needed, if you want to try out local changes/customizations to that Angular UI. Otherwise, if you just want to run the default Angular UI, you can skip this step entirely.):

    ```
    docker-compose -f docker/docker-compose.yml build
    ```

    c. Finally, start up both the Angular UI and REST API via Docker:

    ```
    docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml up -d
    ```

        i. If you'd like to monitor the startup process, you can "tail" the logs using "logs -f":

        ```
        docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml logs -f
        ```

        ii. If anything goes wrong, occasionally a simple restart of the images will resolve it. Just do a "down" followed by an "up -d" again:

        ```
        # Shut down everything
        docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml down
        # Restart everything
        docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml up -d
        ```

4. At this point, you should be able to see a completely *empty* DSpace site. You may now choose to either add test/demo content (see step #5 below) and/or add an initial Administrator account (see step #6 below).
    a. User Interface: http://localhost:4000/
    b. REST API: http://localhost:8080/server/
5. Next, optionally, you can **add test data to your Docker instance**. We have two sets of test data available depending on what you want to test out. CHOOSE ONE.
    a. *[Option #1: Use AIP test data]* We have a set of AIP (Archival Information Package) data which was exported from a DSpace 6.x instance. It's also the easiest to quickly import as we have an ingest script written for Docker using the "dspace-cli" container. Just run:

    ```
    # If you don't have an Admin created with the email "test@test.edu", create it. The AIP ingest
    runs as that user by default.
    docker-compose -p d8 -f docker/cli.yml run --rm dspace-cli create-administrator -e test@test.edu -
    f admin -l user -p admin -c en

    # This second command will import a batch of test/sample AIPs (see "cli.ingest.yml" for more info)
    docker-compose -p d8 -f docker/cli.yml -f ./docker/cli.ingest.yml run --rm dspace-cli
    ```

    b. *[Option #2: Use a database dump of Entities test data]* Alternatively, if you'd like to instead test the new Configurable Entities features, we have a separate database dump which provides Entity test data. (This test data is not yet available in AIP format). Here's how you'd switch your Docker instance to using the Configurable Entities test data
        i. First, you unfortunately need to completely shut down any running volumes and remove them. We will be replacing them with a database dump of Entity test data.

```
# Shut down the running containers
docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml down

# Remove ALL existing volumes
docker volume rm $(docker volume ls -q)

# NOTE: If you don't want to remove all volumes, you can also remove the volumes in two
steps
# First, list all volumes, and look for any that have a name starting with "d7". There
likely will be 4.
docker volume ls
# Then, delete each volume (by name) one by one.
docker volume rm [volume-name]
```

ii. Now, let's recreate those containers with the Configurable Entities test data included:

```
# NOTE: the `db.entities.yml` here will startup a database image with Entities test data
included (from a database dump)
docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml -f
docker/db.entities.yml up -d

# Optionally watch the logs of that command to make sure everything starts back up properly
docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml -f
docker/db.entities.yml logs -f

# Finally, once started, also load up the Entities test "assetstore" (files) & trigger a
reindex
docker-compose -p d8 -f docker/cli.yml -f docker/cli.assetstore.yml run --rm dspace-cli
```

6. Finally, in order to have an initial login, let's create an **initial Administrator account** using the "dspace-cli" container:

```
# This example creates an Admin user with email "test@test.edu" and password "admin".
# (You may have already created this user above, before loading AIP test data. If so, you can skip this)
docker-compose -p d8 -f docker/cli.yml run --rm dspace-cli create-administrator -e test@test.edu -f
admin -l user -p admin -c en
```

7. After a few minutes, you should have a full local installation of DSpace 8 Preview (with test data).
    a. User Interface: http://localhost:4000/
    b. REST API: http://localhost:8080/server/
    c. Admin Login: (whatever login you setup via the "dspace-cli" command above)
8. Objects/Pages of interest (within the test data):
    a. Journal Example represents a journal with journal volumes, issues and articles, as detailed in the Configurable Entities Design.
    b. Publications Example contains publications which contain a combination of plain-text authors and related author entities. It also contains relations to Research Projects and Organizational Units, as detailed in the Configurable Entities design. Navigating to e.g. a Person will reveal their relations to Publications, Research Projects and Organizational Units.
    c. Submission/Workflow functionality can be tested using one of the collections in this community where the workflow is enabled. You can use the accounts mentioned above to perform the submission and workflow steps. The admin account can perform the submission and all workflow steps as well.
    d. Once logged in, MyDSpace functionality is found in the user menu (upper right). Submissions can be started from that page, or via the "New Item" admin menu (if logged in as an Admin).
9. Once you are done testing, you can stop Docker and clean up the data (deleting the volumes).

```
# Shut down the running containers
docker-compose -p d8 -f docker/docker-compose.yml -f docker/docker-compose-rest.yml down

# Remove all volumes, images, etc (This removes all the existing data and images)
docker system prune --volumes
```

Install the User Interface only

The instructions below are meant as a quick guide for how to install the User Interface quickly & use the REST API demo as a backend. Please keep in mind the official installation instructions are at Installing DSpace. These "user interface only" instructions are only useful for trying out the UI for demo /sandboxing purposes.

Running only the DSpace user interface can be done quickly & easily on any operating system (Windows, Mac OS, or Linux). By default, the installed user interface will use the DSpace REST API Demo (https://sandbox.dspace.org/server/) as its backend. This means that you will immediately see test data (from that demo site) and be able to interact with it. However, you will need to authenticate using the REST API Demo account. Any changes you make will also obviously appear on that shared REST API Demo site.

1. First, you will need to install the necessary prerequisites.
    a. Git is required. If you do not already have it installed (try running `git version` at your command line), you can install it via https://git-scm.com/downloads (or via your local package manager tools for Linux or Mac OS)
    b. Node.js (v18 or v20) is required.
        i. NPM (v8.x or above) is required. Don't worry, this gets installed with Node.js though, so you don't need to do anything extra here.
    c. Yarn (v1.x) package manage. Just install the latest stable version of 1.x
2. Then, download our "dspace-angular" application (which is the DSpace user interface) and run it. All you should need to do is the steps in the "Quick Start" at: https://github.com/DSpace/dspace-angular/#quick-start

```
# clone the repo
git clone https://github.com/DSpace/dspace-angular.git

# change directory to our repo
cd dspace-angular

# Switch to the main branch (pre-8.0 code)
git checkout main

# install the local dependencies
yarn install

# start the server
yarn start
```

3. After a few minutes, the user interface will be running on your local machine. Again, it will be accessing the REST API Demo site, so a stable internet connection is required.
    a. User Interface: http://localhost:4000/
    b. REST API (remote demo site): https://sandbox.dspace.org/server/
    c. Admin Login: dspacedemo+admin@gmail.com, Password: dspace
    d. Submitter Login: dspacedemo+submit@gmail.com, Password: dspace
4. Objects/Pages of interest (within the test data):
    a. Journal Example represents a journal with journal volumes, issues and articles, as detailed in the Configurable Entities Design.
    b. Publications Example contains publications which contain a combination of plain-text authors and related author entities. It also contains relations to Research Projects and Organizational Units, as detailed in the Configurable Entities design. Navigating to e.g. a Person will reveal their relations to Publications, Research Projects and Organizational Units.
    c. Submission/Workflow functionality can be tested using one of the collections in this community where the workflow is enabled. You can use the accounts mentioned above to perform the submission and workflow steps. The admin account can perform the submission and all workflow steps as well.
    d. Once logged in, MyDSpace functionality is found in the user menu (upper right). Submissions can be started from that page, or via the "New Item" admin menu (if logged in as an Admin)
5. One you are done testing, you can stop the locally running DSpace 7 user interface via `Ctrl+C`

Install both the User Interface and REST API locally

The DSpace 8 installation guide describes manually installing both the frontend (User Interface) and backend (REST API) of DSpace on a local machine. *T he frontend and backend need not be installed on the same machine.*

Reporting Issues

If you discover an issue in DSpace, here's how to report it:

- First, if you aren't sure whether it's a bug or want advice, feel free to ask on Slack or use one of our other available Support options.
    - On Slack, the DSpace development team uses the #angular-ui channel (for UI discussions) and #rest-api channel (for REST API discussions). Though even reporting it on #dev (general developer channel) is perfectly fine.
- If you are sure it's an issue, **create an issue ticket**
    - Frontend / User Interface issues (including accessibility issues) can be reported via GitHub Issues at https://github.com/DSpace/dspace-angular/issues
    - Backend / REST API issues should be reported via GitHub issues at https://github.com/DSpace/DSpace/issues
    - (If you aren't sure where to report the issue, don't worry. Just report it via either route, and we'll move it to the proper place as needed.)
- Once it is reported, we'll analyze it and schedule it to be completed (based on priority). Please make sure to note how you found this issue and/or any steps to reproduce it.

Want to help out?

Thank you first for trying out DSpace ! Reporting issues (see above) or simply trying things out is already a big help. However, there are other ways you can contribute and help make DSpace even better (and/or help it get released even more quickly):

- **Translate DSpace**. If you know another language (or two or three) and want to help us with our translations, see DSpace 7 Translation - Internationalization (i18n) - Localization (l10n)
- **Help us improve the DSpace Documentation.** While we do our best to get our development team to write some basic Documentation for every new feature, we know that documentation is sometimes a bit technical or not as user friendly as it could be. (No offense to our developers, they are doing what they do best, help us build great new features!) If you are interested in helping us improve our Documentation, see DSpace 8 Documentation. We especially could use help writing new end user documentation (e.g. how to do ___ in the DSpace 7 user interface), as we have a brand new user interface!
- **Help us fix bugs or review new features as they are built.** If you are a bit more technical, we could use more developers willing to chip in on small bugs and/or do some quick testing (or review) of new GitHub Pull Requests. See Testing DSpace 7 Pull Requests