Testing DSpace Github Pull Requests

This documentation is meant to show how to set up your computer to test DSpace Github Pull Requests (PRs). PRs are codes including proposal for new features or bug corrections.

Work in progress

This page is still being continuously updated and improved. It is a work in progress.

Getting ready

Step 1: Install required software

To start working on PR review and testing, the following applications are needed:

- GitHub login. This lets you add a comment to any PR you test, letting developers know if you found problems or if all your tests were successful.
- GIT [https://git-scm.com/downloads](https://git-scm.com/downloads). This makes it easier to "checkout" (i.e. download) the PR code so you can test it.
- Install GitHub CLI. This makes it easier to "checkout" (i.e. download) the PR code so you can test it.
- Install Docker Desktop. This is the easiest way to get a DSpace backend installed locally for testing new (frontend or backend) PRs.
  - Windows 11: ensure Hyper-V are turned on in the Windows Features (TO ADD). May require administrator rights on the computer.
  - Once installed, you will be required to restart your computer. When first launched, an error may appear mentioning that WSL for Windows is needed. If so, install WSL. When installing WSL you will be asked for a Linux Distro.
  - **NOTE:** Docker recently changed their service agreement, but Docker remains free for personal use or non-commercial open source projects like DSpace.
  - **NOTE for Windows 10:** Docker Desktop can sometimes become a bit "memory-hungry" when it is running. It's best to either disable the "Start Docker Desktop when you log in" setting (so that it only runs when needed). Alternatively, if you are using the default WSL2 backend, you can create a ".wslconfig" file in your user directory and limit the amount of memory you let Docker use (at least 4-8GB is recommended if you can spare it):

```
.wslconfig
[ws12]
memory=8GB
```

Congratulations! You have installed the required software!

Step 2: Set up DSpace Docker

This step must be followed after all software listed in step one are installed. For additional details and more up-to-date information on DSpace Docker files and configurations, refer to the documentation on DSpace Github (Docker Compose files for DSpace Backend and Angular Docker Compose files)

GitHub CLI authentication

This operation is done once. It is must be done before beginning to install either DSpace Backend or Frontend.

<table>
<thead>
<tr>
<th>Actions</th>
<th>Command line</th>
<th>Purpose</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>GitHub CLI (gh) authentication</td>
<td>gh auth login</td>
<td>Git authentication using github credentials Needed to use Git command. If this is your first time using GitHub CLI (gh), you likely will need to authenticate with it using your GitHub login.</td>
<td>Git</td>
</tr>
</tbody>
</table>

Also possible to use Git Bash:

1. Install Git Bash & verify the installation by running "git version" from Command Prompt or Powershell
2. No need to authenticate with Github credentials.

Installing Backend

3-step process (details in table below):

1. Cloning the GitHub repository
2. Getting the DSpace Backend images from the Docker Registry
## 3. Starting the Docker Backend Containers

<table>
<thead>
<tr>
<th>Actions</th>
<th>Command line</th>
<th>Purpose</th>
<th>Software</th>
</tr>
</thead>
</table>
| Clone the backend codebase | # Run from command-line where ever you want the "DSpace" directory of code to be created.  
gh repo clone DSpace/DSpace  
# Once downloaded, change into that directory  
cd DSpace | Before cloning carefully select your parent folder.  
Getting all the files needed to run DSpace Backend.  
This downloads all the code from https://github.com/DSpace/DSpace into a local directory named “DSpace”. | Git |
| Start Docker Desktop | If it is not already running, start Docker Desktop on your machine. It MUST be running for any “docker-compose” or “docker” commands to work.  
a. On Windows, if Docker Desktop is running, you'll see the Docker whale logo in your taskbar's notification area. | Docker |
| Pull down the latest backend images | # This command should be run from the "DSpace" code directory  
docker-compose -f docker-compose.yml -f docker-compose-cli.yml pull | Pull down the latest backend images from DockerHub. This downloads all the prebuilt Docker images so you don't need to rebuild them locally. | Docker command line |
| Start the Docker DSpace Backend Containers (& automatically install our Entities Test Data) | # First start the backend with the test data  
docker-compose -p d7 -f docker-compose.yml -f dspace/src/main/docker-compose/db.entities.yml up -d  
# [OPTIONNAL] Run "logs -f" to watch the logs for everything to start up.  
docker-compose -p d7 -f docker-compose.yml -f dspace/src/main/docker-compose/db.entities.yml logs -f  
# (Click Ctrl+C to exit logs view) | This command creates the 3 Docker containers for the Backend (dspace db for the database ; dspace solr for the solr indexes ; dspace for the backend server ; angular for the frontend UI) and then starts them up.  
Warning: dspace db is the only Docker container that is not started automatically (the two others, dspace solr and dspace, are)  
The backend will be started with the test data and may take a few minutes. It should be completed before moving to the next action.  
# Wait a few minutes until http://localhost:8080/server/ responds. | Docker command line |
| Download, install and reindex the test data assetstore (of files) | # Finally, download the test data assetstore (of files) and install it, reindexing all content  
docker-compose -p d7 -f docker-compose-cli.yml -f dspace/src/main/docker-compose/cli.assetstore.yml run dspace-cli | | Docker command line |
| Check system is up and running | DONE!  
At this point, you should be able to go to http://localhost:8080/server/ and see a backend similar to https://api7.dspace.org/server/  
a. You should also see some test data in that backend.  
For example, clicking on the "collections" endpoint should return some Collections: http://localhost:8080/server/http://localhost:8080/server/api/core/collections | Web browser (http://localhost:8080/server/)  
The Hal Browser page is displayed |
## Installing Frontend

3 step process (details in table below):

1. Cloning the GitHub repository
2. Getting the DSpace FrontEnd images from the Docker Registry
3. Starting the Docker DSpace Frontend Containers

<table>
<thead>
<tr>
<th>Actions</th>
<th>Command line</th>
<th>Purpose</th>
<th>Software</th>
</tr>
</thead>
</table>
| Clone the frontend codebase | # Run from command-line where ever you want the "dspace-angular" directory of code to be created. 
  gh repo clone DSpace/dspace-angular 
  # Once downloaded, change into that directory 
  cd dspace-angular | Getting all files needed to run DSpace frontend (Angular) 
  First, clone the frontend codebase using GitHub CLI (gh). This downloads all the code from https://github.com/DSpace/dspace-angular to a local directory named "dspace-angular". You may wish to run this command from the same parent directory where you cloned the backend (just to keep this new "dspace-angular" folder next to the "DSpace" folder where the backend code resides). | Git |
| Ensure Docker Desktop is running | | If it is not already running, start Docker Desktop on your machine. It MUST be running for any "docker-compose" or "docker" commands to work. 
  On Windows, if Docker Desktop is running, you'll see the Docker whale logo in your taskbar's notification area. Using Docker, pull down the latest frontend images from DockerHub. This downloads all the prebuilt Docker images so you don't need to rebuild them locally. | Docker command line |
| Pull down the latest frontend images | # This command should be run from the "dspace-angular" code directory 
  docker-compose -f docker-compose.yml pull | Using Docker, pull down the latest frontend images from DockerHub. This downloads all the prebuilt Docker images so you don't need to rebuild them locally. | Docker command line |
| Start Docker Container for DSpace frontend/UI | docker-compose -p d7 -f docker/docker-compose.yml up -d 
  # Optionally run "logs -f" to watch the logs for everything to start up. 
  docker-compose -p d7 -f docker/docker-compose.yml logs -f 
  # (Click Ctrl+C to exit logs view) | This command will create the Docker Container for the Frontend and then start it up. | Docker command line |
| Check UI is up and running | 1. DONE! In a few minutes, the User Interface should be available at http://localhost:4000/. It should automatically be pointed at your Backend (also running on Docker)! 
  a. Test it out by logging in using one of the demo accounts! Login: dspacedemo+admin@gmail.com . Password: dspace | | Web Browser (http://localhost:4000/) |

**Step 3: Ensure everything is up and running**
Checking Docker Desktop Containers

Your Docker Desktop Containers page should look like the image below, with the d7 project containing at least 4 containers: dspacedb for the database; dspacesolr for the solr indexes; dspace for the backend server; angular for the frontend UI.

<table>
<thead>
<tr>
<th>Name</th>
<th>Image</th>
<th>Status</th>
<th>CPU (%)</th>
<th>Port(s)</th>
<th>Last started</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>dspace-cli-nan-cc728af621ad</td>
<td>dspace/dspace-cli/latest</td>
<td>Exited</td>
<td>0%</td>
<td></td>
<td>6 minutes ago</td>
<td>▶️</td>
</tr>
<tr>
<td>dspace</td>
<td>dspace/dspace:latest</td>
<td>Running</td>
<td>175.7%</td>
<td></td>
<td>10 seconds ago</td>
<td>▶️</td>
</tr>
<tr>
<td>dspacedb</td>
<td>dspace/dspace-postgres-ecryptoserver:latest</td>
<td>Running</td>
<td>0.25%</td>
<td>8000-8001</td>
<td>10 minutes ago</td>
<td>▶️</td>
</tr>
<tr>
<td>dspace solr</td>
<td>dspace/dspace-solr:latest</td>
<td>Running</td>
<td>0.73%</td>
<td>5432-5433</td>
<td>8 minutes ago</td>
<td>▶️</td>
</tr>
<tr>
<td>dspace- angular</td>
<td>dspace/dspace-angular:latest</td>
<td>Running</td>
<td>174.72%</td>
<td>4000-4000</td>
<td>10 seconds ago</td>
<td>▶️</td>
</tr>
</tbody>
</table>

TODO: ALSO ADD PRINT SCREEN OF UI AND BACKEND

Checking DSpace URL

Once you start the containers either through command line or Docker UI, the following URLs should be active.

- **Backend:** http://localhost:8080/server/
- **Angular UI:** http://localhost:4000/home
- **OAI:** http://localhost:8080/server/oai/request?verb=Identify

Starting Testing PRs

How to find PR for testing

If you are less technical, or just getting started, we recommend beginning your testing (or code review) with Pull Requests that contain the "1 Approval" label. These are smaller Pull Requests that need a very quick test or review (usually a small bug fix or improvement).

- Backend pull requests with "1 Approval" label
- Frontend pull requests with "1 Approval" label

You are welcome to test any open pull request. Even if the pull request is already assigned to other testers, it can help speed up its completion / acceptance if others also test it.

We recommend testing whichever pull requests are of interest to you. You may find one relating to a bug that you'd also seen, or something that sounds easy for you to test, etc. The choice is up to you.

How to select a PR to test

Things important to consider before selecting a issue to test:

- No merge conflict detected (merge conflict tag)
- No changes have been requested by a reviewer (changes requested tags)
- No "work in progress" tag
- All automated test have passed without error
- Looking for an easy one? Look for the "One approval" tag.

Understanding what to test

For testing:

- Look if a Github Issue is linked to the PR, it usually gives more information on the feature to test.
- If test instructions are unclear, put a comment asking for instructions
- Tester do not need to be assign a PR to test it.
If the PR needs a specific configuration, it can be changed: recommendation is to use a local.cfg containing the config value for the test. i.e., if the feature change in the PR use emails, it is necessary to configure SMTP using your SMTP reference.

Testing the DSpace Pull Request (PR)

<table>
<thead>
<tr>
<th>Move to relevant directory</th>
<th>Testing a Frontend PR</th>
<th>Testing a Backend PR</th>
</tr>
</thead>
<tbody>
<tr>
<td>dspace-angular</td>
<td>DSpace</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Checkout the PR code</th>
<th>gh pr checkout 1234 (where 1234 is the PR number)</th>
<th>gh pr checkout 1234 (where 1234 is the PR number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>dspace-angular</td>
<td>DSpace</td>
<td>DSpace</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rebuild based on that PR's code</th>
<th>docker-compose -p d7 -f docker/docker-compose.yml build</th>
<th>docker-compose -f docker-compose.yml -f docker-compose-cli.yml build</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Note: Build duration (Angular): 3-4 minutes with a good bandwidth/connexion</td>
<td>Note: Build duration (Backend): environ 5-10 minutes with a good bandwidth/connexion</td>
</tr>
<tr>
<td></td>
<td>This will create a new Docker Image including the PR code.</td>
<td>This will create a new Docker Image including the PR code.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start services needed</th>
<th>Frontend can be start using command line or Docker Software UI.</th>
<th>Backend can be start using command line or Docker Software UI.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commandline:</td>
<td>Ensure you are positionned in dSpace-angular folder (cd command to dspace-angular)</td>
<td>Ensure you are positionned in DSpace folder (cd to DSpace if not)</td>
</tr>
<tr>
<td>Run following command:</td>
<td>docker-compose -p d7 -f docker/docker-compose.yml up -d</td>
<td>docker-compose -p d7 up -d</td>
</tr>
<tr>
<td></td>
<td>This command will start the container using the new Docker image created.</td>
<td>This command will start the container using the new Docker image created.</td>
</tr>
</tbody>
</table>

Once you are ready to check the PRs: open Dev tool of your preferred browser

If an error occurs, check the log, either backend or frontend log can be accessed using Docker by clicking on the containers Name (i.e: dspace-angular or dspace-db, etc.)

Reporting test results

The PR's description should describe how to test it and what you should see.

a. If you notice any odd behavior, check for errors in the UI using your browser's DevTools. See Troubleshoot an error#FindingtheErrorMessageintheUserInterface

b. Based on what you find, add a comment to the PR. If it works, congratulate the developer. If it doesn't, let them know the error you saw (and what you clicked on when that error occurred).