Argovis Docker Build Instructions

Tyler Tucker, Donata Giglio, Megan Scanderbeg

Scope

This document objective is to show a user familiar with Linux OS, Git, Docker and Python to build the Argovis database using a small subset of Argo data.

Docker and Docker-Compose

Docker builds Argovis and its dependencies into separate containers, exposing ports and shared volumes as needed. Docker-compose runs these containers and performs the setup. It is assumed that the user has Docker and Docker-Compose installed on their machine, and they are able to create containers.

Instructions

These instructions will have you have your very own local database serving a subset of argo profiles stored on a MongoDB database.

1. Download the repositories from GitHub
2. Build and run containers
3. Create a directory of .nc files
4. Convert .nc files to profile documents stored in the database
5. Check for profiles on argovis front-end

Download the repositories from GitHub

Repositories are found on GitHub, there are several branches, with master as the default.

1. Using a terminal, make a directory called argovis-tutorial. Enter directory
   ```
   mkdir argovis-tutorial
   cd argovis-tutorial
   ```
2. Clone the following repositories
   ```
   git clone --single-branch --branch dev --depth=1
   https://github.com/tylertucker202/argo-database.git
   git clone --single-branch --branch dev --depth=1
   https://github.com/tylertucker202/argovisNg.git
   ```
3. Enter argovisNg and clone the additional repository
   ```
   cd argovisNg
   ```

```
git clone --single-branch --branch dev --depth=1
https://github.com/tylertucker202/argo.git
```

Build and run containers

Docker and Docker-compose automate the build process for the database, front-end, back-end components. It creates containers with the necessary software and dependencies. The instructions to Docker is located in the docker-compose.yml file in argovisNg directory.

1. Make another directory for the mongDB database
   Mkdir mongodb.
2. In your favorite text editor, set the mongoDB directory path in the docker-compose.yml file. The path is in the second to last line in the image below.

```
version: '3'
services:
 argo-express:
   build: . #specify the directory of the Dockerfile
   restart: always
   ports:
   - '3000:3000' # specify port forwarding
   links:
   - database
 database:
   image: mongo:4.2.3  # specify image to build container from
   restart: always
   ports:
     - '127.0.0.1:27017:27017' # include localhost ip to restrict access
to localhost
   volumes:
     - /data/argovis/storage/mongodb:/data/db:Z # production
```

3. While in the argovisNg directory, run the following command
   `Docker-compose build && Docker-compose up &`
   You will see some text cascade down the terminal. At this point, leave this terminal open and create another terminal instance.
   By the way, you can turn off the components with the command
   `docker-compose down`

   ```
   Docker-compose normally outputs to sdout. To log to a file,
   instead of "docker-compose up &" try "docker-compose up -d" for
   detached mode. Logs are read with the command
   docker-compose logs --tail 10
   ```

```
For the last 10 lines of each container
```

Let’s check if the app is running. Visit the app in your favorite browser http://localhost:3000/ng/home. You should be taken to an instance of Argovis. Note there are no dots. That's because there is nothing in the database at the moment.

Create a directory of argo .nc profiles

The FTP server at ftp://ftp.ifremer.fr/ifremer/argo contains all the Argo profile data used by the database. Getting the data into the database involves downloading files and directories to argo. We will use the rsync utility to download a few profiles, and use the argo-database repository to convert these files to entries on the server.

1. Make a directory in argo-database that will store the profiles.
   ```
   cd argo-database
   mkdir profiles
   mkdir profiles/aoml
   ```
2. Run the rsync command
   ```
   rsync -arvzhim --delete --include='**/'
   --include='**/profiles/[RDM]*.nc' --exclude='*'
   --exclude='**/profiles/B*' vdmzrs.ifremer.fr::argo/aoml/4902911
   ./profiles/aoml
   ```
   All the profiles from the float 4902911 should have been written to profiles/aoml/profiles. You can add more or all of the floats to the directory if you wish.

Rsync can download profiles individually, single out dacs, or entire databases. For this instance, these profiles should be enough.

Convert .nc files to profile documents stored in the database

Data stored in .nc files in transformed into a BSON (BINARY JSON) object and stored in the mongoDB database that is currently running on your PC. The python scripts in argo-database make this happen.

1. While in the the argo-database directory, Build the docker image with the command
   ```
   docker build -t argo-db:tut .
   ```
   Docker build will install python and dependencies to the container.
2. Save the argo-directory absolute path as a variable with the command
   TUTDIR=`pwd`
   Note that I am using the ‘`’ (to the left of the ‘1’ key on your keyboard)
3. Run and enter container with the command:
   docker run -u $(id -u):$(id -g) -it --entrypoint /bin/bash --net=host -v $TUTDIR:/usr/src/argo-database/:Z argo-db:tut
   You are inside the container and can run python. If you need to go back, exiting the container with ctrl+d

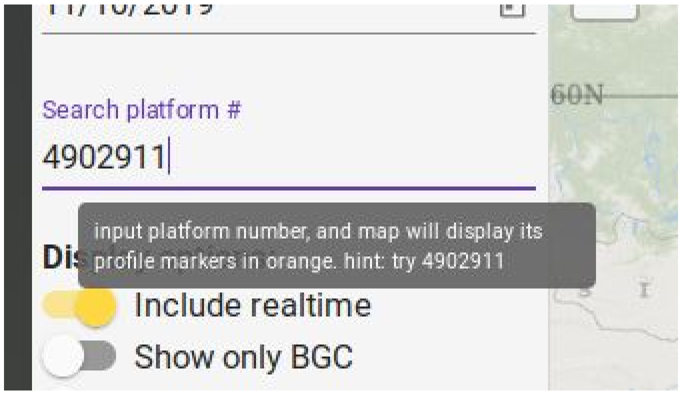
4. Check if container is running with the docker command (in a separate terminal)
   ```
   docker ps --format "table
   {{.Names}}\\t{{.Image}}\\t{{.Status}}"
   ```
   You should see a table similar to this one
   ```
   NAMES                         IMAGE                    STATUS
   sharp_allen                   argo-db:tut              Up 59 seconds
   argovisng_argo-express_1      argovisng_argo-express   Up 2 hours
   argovisng_database_1          mongo                    Up 2 hours
   ```

   You can stop a container by running the command
   ```
   docker stop {image name}
   IE
   docker stop sharp_allen
   ```

5. While in the docker container, navigate to the add-profiles directory and enter the command
   ```
   python add_profiles.py --mirrorDir
   /usr/src/argo-database/profiles --logName tutorial.log
   ```
   You can check the log file named tutorial.log in the same directory for more details. It's helpful to check which profiles have been added and which ones are rejected.

Check for profiles on argovis front-end

Next we want to check if the profiles have been added to the database by searching for the platform on your local version of argovis.
Navigate to the instance of argovis running on your browser. In the platform selection, enter the platform number belonging to the profiles we just added (4902911)


Search platform #
4902911
input platform number, and map will display its profile markers in orange. hint: try 4902911

The profiles belonging to this item should appear in the Atlantic ocean

Hudson Bay
Labrador Basin
Labrador Sea
NORTH ATLANTIC OCEAN
NORTH AMERICA
Mid-Atlantic Ridge
Gulf of Mexico
Sargasso Sea
Caribbean Sea

Conclusion:

You have an instance of argovis running. Note that everything is contained in the tutorial-directory. Adding too many profiles can lead to the hard disk filling up. Be aware that the full database is around 16 GB, and the entire Argo mirror is well over 100GB. Also keep in mind MongoDB is designed to run on an XFS file system, which may need its own separate HD partition.