Open Education - developer's guide

Introduction

This project is an initiative towards making education open source, and to achieve highly interactive teaching and presentation material.

- Prof. Jonathan Appavoo's talk https://research.redhat.com/events/steps-toward-open-source-education/

Implementation

From the implementation perspective, the project has two significant components.

Infrastructure/templates

Provide required support for the textbook content, building source-to-image builder image/templates for the jupyter notebooks.

- https://github.com/OPEFFORT/ope/tree/container-redhat
- https://github.com/jappavoo/UndertheCovers/tree/container

Textbook content

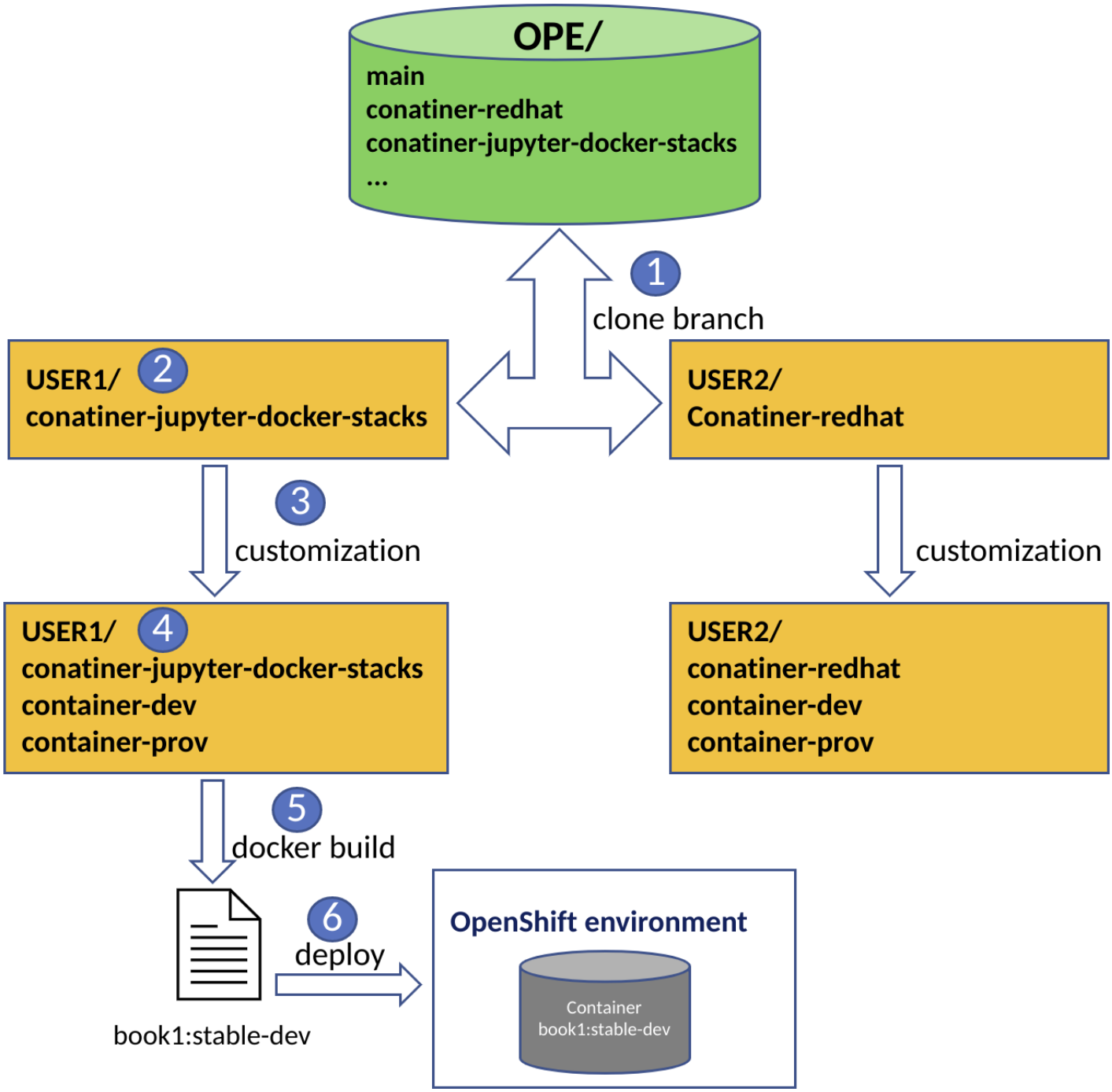
Actual teaching material based on JupyterBook with rich presentations etc.

- https://github.com/jappavoo/UndertheCovers/tree/main
- https://github.com/okrieg/openos

Developing a new textbook template - stages

OPE repo contains the fedora-based and ubuntu based templates as branches for the jupyter notebook builder-image. This is the starting point and the subsequent steps are as below:

1. Choose the required base template i.e fedora or ubuntu based images. Clone the respective branch to your local repository.
2. Update configurations related to target image registry, docker image tags etc. Add new layers to the Dockerfile to include additional libraries and other configurations to meet the requirements.
3. Create new branches for every deploying environment and update the **ope_uid** file with that of the environment user id.
4. Build the docker image with the respective customization (CUST) option. Export to docker image registries and make the repository publicly visible.
5. Add the new image details in the OpenShift environment and deploy.



Jupyter notebook templates

We have created a fedora **(container-redhat)** and an ubuntu based **(container-jupyter-docker-stacks)** jupyter notebook template. The container template docker image is built over two stages to achieve better organization and to reduce size.

The dockerfile of these templates includes the two stages as:

```
#################### STAGE ONE ######################
ARG arg1
ARG arg2
.
.
ARG argn

FROM base_image:tag AS stage-one

RUN layer1
.
.
RUN layern

#################### STAGE TWO ######################

ARG arg1
.
.
ARG argn

FROM base_image:tag AS stage-two

COPY --from=stage-one <source_path> <dest_path>

RUN layer1
.
.
RUN layern
```

Fedora based template

Source code base - https://github.com/AbiShanna/ope/tree/container-redhat

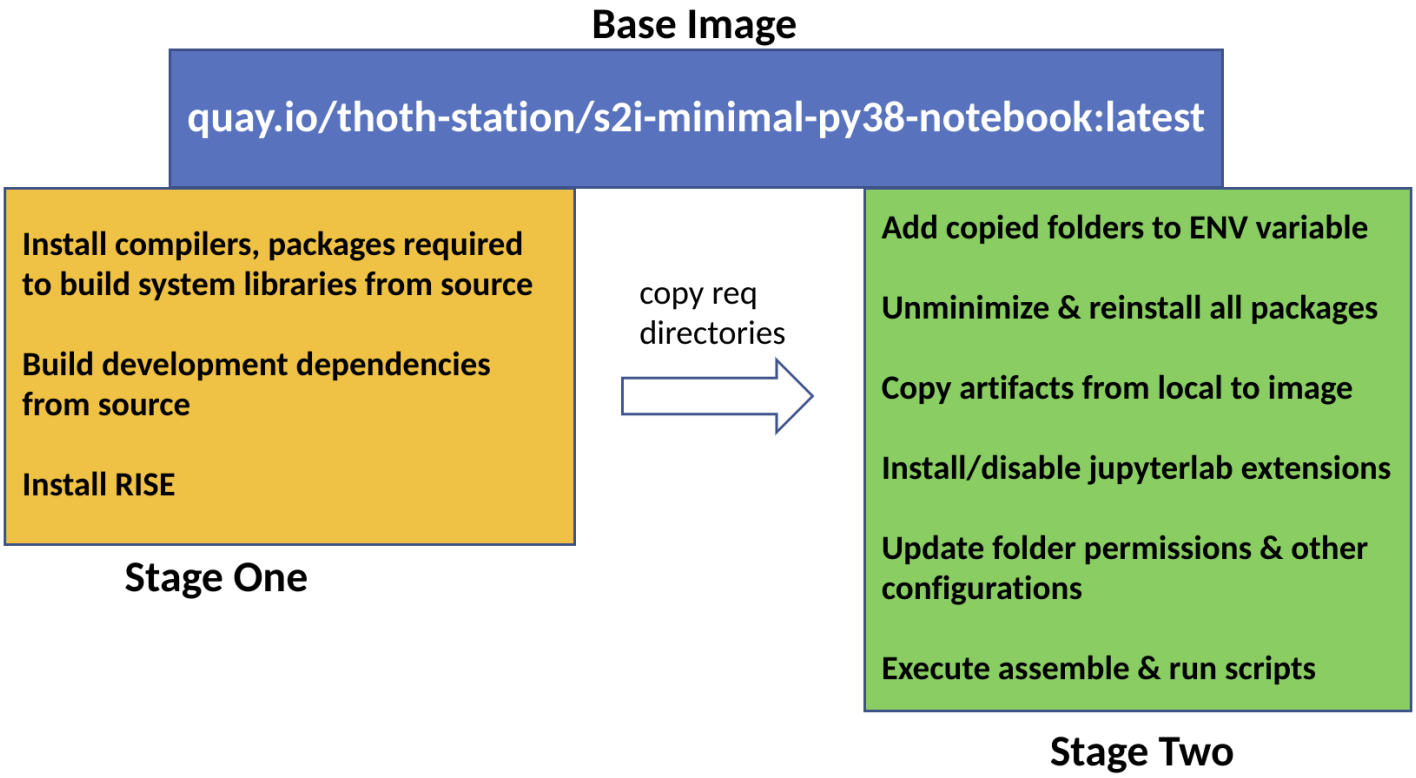
This generic template is built using Red Hat's Univerbal Base Image (UBI). The base image s2i-minimal-jupyter-notebook is customized with multiple layers of softwares to support author, build and publish textbooks, extensions to notebook and configurations that can be deployed in an OpenShift managed cloud environment.

To customize and author materials, one can clone the OPE/container branch and can from there create more branches based on the local repo for each
environment like development, testing, production etc.

Ubuntu based template

Source code base : https://github.com/AbiShanna/UndertheCovers/tree/container.

The ubuntu based template is created using the open source Jupyter Book project and the executablebooks/cookiecutter-jupyter-book template. The below diagram represents the stages for the ubuntu-based jupyter template https://github.com/OPEFFORT/ope/tree/container-redhat



Understanding the artifacts involved in the build step

Both the fedora and the ubuntu based repository has the following folder structure.

The build command invokes the actual docker build process with a list of arguments passed from multiple configuration files. The assemble and run scripts are from the s2i-minimal-notebook image of thoth-station. The assemble script installs the provided list of python libraries, basic jupyterlab extensions and the run script is executed during the run time which creates the user account, sets the required folder structure etc.

Notes specific to fedora image:

- Source-to-image:

  Source-to-image is a framework for building reproducible container images from source code. S2I produces ready-to-run images by injecting source code into a container image and creating a new image that runs the assembled application.

The jupyter notebook templates created will be ingested with the textbook content and the final containers will be made available for the students to use.

Reference:
https://github.com/openshift/source-to-image
https://docs.openshift.com/container-platform/3.11/dev_guide/migrating_applications/S2I_tool.html

- These images are aimed to be deployed in an OpenShift managed cloud environment. The default home directory of the s2i images will be '/opt/app-root/src'. The persistent volume provided by the environment is also by default mapped to the same '/opt/app-root/src' directory.

- Understanding micropipenv:

  Unlike the Ubuntu based image, where we list the python libraries with versions to be installed, the s2i process utilizes a new wrapper tool called *micropipenv* to handle the installations. 'Pipfile', 'Pipfile.lock' is parsed by Thamos tool. After checking it installs the listed libraries.

- Understanding the Thamos tool:
  Project Thoth of Red Hat can resolve python software packages to the "greatest" library version. Thoth offers a recommendation based on security, latest versions, performance etc, to determine the "greatest" version based on different criteria, which guides the installation process. Thoth periodically fetches the database of known vulnerabilities and automatically blocks the resolution of software package versions that are prone to security vulnerabilities.

  Thamos is a command-line interface tool to connect with thoth and to perform the static source code analysis. Thamos cli is available in the s2i images by default.

  Reference:
  https://www.youtube.com/watch?v=NvURYXd2Oe4&t=1120s

- Understanding the 'generate_container_user' script present under *'/opt/app-root/etc'*

During the execution of 'run' script, if the current user is not root/default(1001), this script creates an entry to /etc/passwd with no home directory and sets the required permissions.

- Assemble and run scripts:

Notes specific to Ubuntu image:

- Jovyan user:
  The default user of the docker image is jovyan with uid 1000. During build time, we update this uid with the base/ope_uid to match the target environment user id.

- Different persistent volume mapping:
  The base image is container-jupyter-docker-stacks which is not of s2i format. The default home directory is /home/jovyan which is different from the OpenShift's /opt/app-root/src'. The start-notebook.sh script of the jupyter stacks is modified to handle the persistent volume mapping to the user home directory.

- start-notebook.sh:

-

Other resources

- Open Education (OPE) repository - https://github.com/OPEFFORT/ope
- Open Education Documentation repository - https://github.com/AbiShanna/Ope-Documentation
- Underthecovers ubuntu source - https://github.com/jappavoo/UndertheCovers/tree/container
- BU CAS CS210 course material repository of Prof. Jonathan Appavoo - https://github.com/jappavoo/UndertheCovers
- Basic docker commands:
  - To authenticate quay.io :
    docker login quay.io

  - To get the size consumed by each layer of the docker image:

    docker history --format "{{.ID}}: {{.CreatedBy}}: {{.Size}}" --no-trunc <docker_image>