BLOG POST

BINDERHUB AND JUPYTERLAB

AUTHORS:

Gunjan Lalwani - lalwani.g@husky.neu.edu

Rishabh Jain - jain.rishabh@husky.neu.edu

Ann Sajee - sajee.a@husky.neu.edu

BinderHub

1: Introduction to BinderHub

The primary goal of BinderHub is creating custom computing environments that can be used by many remote users. BinderHub enables an end user to easily specify a desired computing environment from a GitHub repo. BinderHub then serves the custom computing environment at a URL which users can access remotely.

1.1: What is BinderHub?

Allows you to BUILD and REGISTER a Docker image using a GitHub repository

CONNECT with JupyterHub, allowing you to create a public IP address that allows users to interact with the code and environment within a live JupyterHub instance

1.2: BinderHub ties together:

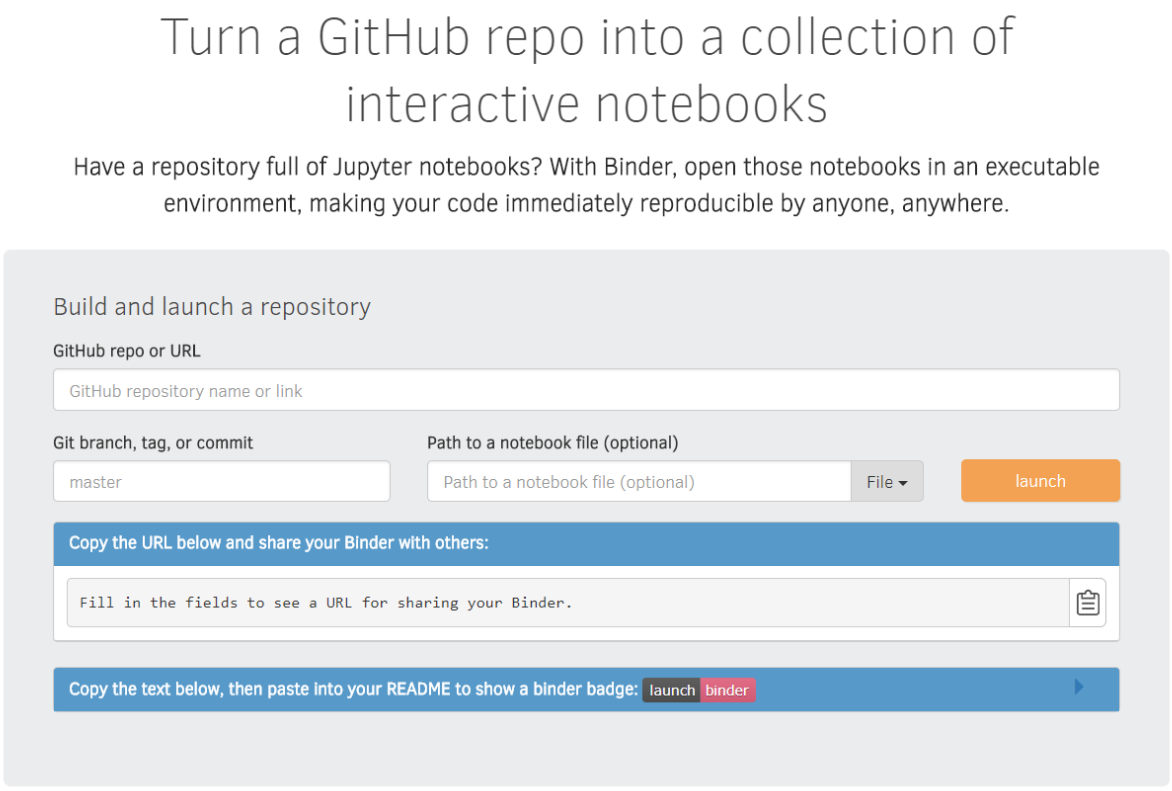
JupyterHub to provide a scalable system for authenticating users and spawning single user Jupyter Notebook servers

Repo2Docker which generates a Docker image using a Git repository hosted online

1.3: Installation:

pip install git+https://github.com/jupyterhub/binderhub

You can also use https://mybinder.org/ to create the docker image of your gtihub repository.



2: The BinderHub Architecture

Below is a high-level overview of the technical pieces that make up a BinderHub deployment.

2.1: Tools used by BinderHub

BinderHub connects several services together to provide on-the-fly creation and registry of Docker images. It utilizes the following tools:

A **cloud provider** such Google Cloud, Microsoft Azure, Amazon EC2, and others

Kubernetes to manage resources on the cloud

Helm to configure and control Kubernetes

Docker to use containers that standardize computing environments

A **BinderHub UI** that users can access to specify GitHub repos they want built

BinderHub to generate Docker images using the URL of a GitHub repository

A **Docker registry** (such as gcr.io) that hosts container images

JupyterHub to deploy temporary containers for users

2.2: What happens when a user clicks a Binder link?

After a user clicks a Binder link, the following chain of events happens:

BinderHub resolves the link to the repository.

BinderHub determines whether a Docker image already exists for the repository at the latestref (git commit hash, branch, or tag).

If the image doesn't exist, BinderHub creates a build pod that uses repo2docker to do the following:

Fetch the repository associated with the link

Build a Docker container image containing the environment specified in configuration files in the repository.

Push that image to a Docker registry, and send the registry information to the BinderHub for future reference.

BinderHub sends the Docker image registry to **JupyterHub**.

JupyterHub creates a Kubernetes pod for the user that serves the built Docker image for the repository.

JupyterHub monitors the user's pod for activity and destroys it after a short period of inactivity.

JupyterLab

1: Introduction to JupyterLab

- An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
- JupyterLab is the next-generation user interface for Project Jupyter. It offers all the familiar building blocks of the classic Jupyter Notebook (notebook, terminal, text editor, file browser, rich outputs, etc.) in a flexible and powerful user inteface. Eventually, JupyterLab will replace the classic Jupyter Notebook after JupyterLab reaches 1.0.
- JupyterLab can be extended using extensions that are npm packages and use our public APIs. You can search for the GitHub topic jupyterlab-extension to find extensions

1.2: Installation

You can install JupyterLab using conda, pip, or pipenv.

conda

If you use conda, you can install as:

```
jupyter serverextension enable --py jupyterlab --sys-prefix
```

pip

If you use pip, you can install it as:

```
pip install jupyterlab
```

If installing using pip install --user, you must add the user-level bin directory to your PATH environment variable to launch jupyter lab.

1.3: Prerequisites and Supported Browsers

Jupyter notebook version 4.3 or later. To check the notebook version:

jupyter notebook --version

The latest versions of the following browsers are currently *known to work*:

- Firefox
- Chrome
- Safari

JupyterLab: Integrated Experience