writing-sample

October 13, 2022

1 Make a technical writing sample with Jupyter Notebooks

Jupyter Notebooks can give technical writers a versatile way to share their work. Unlike a static site generator, such as Jekyll, you can host a dynamic Jupyter Notebook demonstrating both your writing and executable code samples or print a static version of your content to PDF or HTML. To make a Jupyter Notebook and host it on Binder requires that you can competently work through a very common workflow for software documentation writers, who frequently use Markdown and Git.

This tutorial covers the steps required to: 1. Set up git and Github. 2. Install Jupyter through the Anaconda distribution package. 3. Create a Hello World Notebook for testing. 4. Use mybinder.org, to create a remote host for your notebooks.

After you successfully host your notebook on Binder, you can download a static copy of your writing sample or share a URL that directly links viewers to a copy with executable code.

This guide is designed for technical writers who have some exposure to Git and GitHub, but might not be experts yet. There are steps that go into specific detail about how to work with either Git using terminal or Visual Studio Code.

1.1 Before you begin

You need a few programs installed before you can start working in Jupyter. For this guide you will use * Git * Visual Studio Code

You also need a GitHub Account and to Log In to Github on VS Code before you begin.

1.2 Install and set up Jupyter Notebooks

First, you need to install and set up Jupyter Notebooks on your Mac. Jupyter Notebooks are automatically installed with in Anaconda, a Python distribution platform that includes Python 3 as well as some other scientific computing software.

1. Install and set up Anaconda and Jupyter by downloading the Anaconda binaries distribution from Anaconda.org. Then follow the installer wizard steps.

Important: By default, this installer modifies your bash profile to activate the base environment of Anaconda3 when your shell starts up. To disable this, choose **Customize** at the **Installation Type** phase, and disable the **Modify PATH** option.

2. After the installer has finished, open a terminal by navigating to **Applications**»**Utilities**»**Terminal** or using Spotlight and check that Anaconda installed correctly by running `conda -version`.
3. Next check that Jupyter is running correctly. By running the command `jupyter -version`.
4. If you have problems, common Anaconda installation issues are resolved in the Troubleshooting Docs and Jupyter Docs.

Note: You can also open Anaconda navigator, which provides a GUI for you to select which tool you want to use. This can be convenient, however, if you open Jupyter Notebooks from the navigator, it automatically opens the server to the directory where Anaconda is installed, which might be a different location from where you want your writing sample repository. It's better to launch Jupyter from a terminal window in VS Code instead of the navigator.

1.3 Set up a new Git repository on Github

Now that you've installed Anaconda, you can create your new Github repository for your writing samples. 1. Log in to Github online 2. Create a new Git repository for your writing sample notebooks from Github.com 3. Go to the main Github page 4. Select **New** repository 5. Fill in the information for your new repository. 6. Make sure your repository is **Public** because Binder will require access to it later. 7. Add a License. The MIT Open Source License has some standard coverage that allows others to use your materials as long as they cite you as the creator. Learn more about Open Source Licenses from Carnegie Mellon University. 8. Select **Create Repository**

1.4 Clone a copy of your new repository to your computer.

Right now, your new repository only exists on your online GitHub account. You need to download it to your computer to work on it in VS Code. Downloading a copy of your repository to your machine is to make a *Clone* of your repository.

1. From your new Repository on Github (https://github.com/your-username/repository-name), select the **Code** button. This gives you a copyable URL you will need later.
2. On your desktop, open a new folder in VS Code. Sometimes when you open VS Code, it might prompt you to create a new folder or it might open in an existing folder with other files.
3. Open a new folder by going to **File** » **Open Folder**.
4. Create a new folder where you want to create your Jupyter Notebooks. This example created one called **writing-sample**.
5. Clone your Git Repository
   - VS Code: Click **Clone Git Repository** and paste the URL you copied from GitHub.
   - Terminal: Open a terminal in VS Code by selecting **Terminal** » **New Terminal** then enter the command `git clone` and paste the repository URL you copied earlier. The full command looks like the following example:
     `git clone https://github.com/lzdanski/test-sample.git`

You created a new repository and downloaded a local copy. Now you can make changes on your computer to the repo and upload them to your public Github Repository. This means you can

share a URL to your writing sample repository and other people can see your technical writing and source code. You can also now use Binder to dynamically host your Jupyter Notebooks from your public repo.

1.5 Try out Jupyter

Now, you can start working in Jupyter. First, test to make sure it installed correctly. 1. In your open terminal window in VS Code, check the Jupyter version `jupyter -version` 2. Serve Jupyter locally by running `jupyter notebook` in your terminal.

This starts the server to display your jupyter Notebooks at http://localhost:8888/, and a browser window opens to display it automatically. You should only see the directory where you downloaded your repository.

Completing this step confirmes that both Jupyter and VS Code are working correctly locally. Now you can start moving files back and forth from the local to your online repository on Github, called the *remote*.

1.6 Update your Readme and sync with Github remote.

You cloned a copy of your remote repo directly to your computer, now make sure that Github has been configured to push changes to the remote.

1. In terminal, enter `git status`
2. Git returns the status as `on branch main` or `on branch master` depending on your repository settings.
3. Create a new branch to update the Readme
   - From terminal: `git checkout -b update-readme`
   - From VS Code UI: **Branch**»**New Branch**, and name it `update-readme`.
4. Open your readme.md file and add a sentence or two, then save your changes.

After you save your changes, there are visual indicators in the VS Code UI to indicate that you have uncommitted changes. This means you have changed files and have only saved those changes to your local machine, on the new branch you created. There isn't a copy of those changes saved remotely yet. The following picture has yellow arrows pointing to the different locations that VS Code highlights the git status of your files.

You can also see that you have uncommitted changes by typing `git status` in the terminal.

5. Commit your changes to `update-readme`
   - From terminal: `git add readme.md` `git commit -m "Update readme"` `git push origin update-readme`
   - From VS Code: Go to the **Source Control** tab. Next, stage your changes by clicking the + Symbol in the Source Control pane. Then, enter a message and select the **Commit** button. Finally **Publish Branch**.
6. Merge your changes into `main`
   - From terminal: Go to the URL in the terminal output. Enter a message in the field and click **Make pull request**

- From Github: First, push your changes to the remote by selecting **Publish Branch**. Next, go to your repo online. There is a banner that highlights your branch has been remotely updated. Click **Compare and make pull request**. Enter a message in the field and click **Make pull request**

7. Update your local copy of main
   - From terminal: `git checkout main  git pull origin main  git branch -d update-readme`
   - From VS Code: Change your branch to main in the UI and then click **Sync changes**.

1.7 Set up the Jupyter Notebook renderer

Now you set up the Jupyter Notebook renderer on a new branch. 1. Make a new branch * From terminal: `git checkout -b build-sample` * From VS Code: Open the **More options** menu in the Source Control view by selecting **...>Branch>Create branch**. Name your new branch `build-sample` 2. Create a Jupyter Notebook by selecting **File»New file** and choosing **Jupyter Notebook** from the menu of options.

Whether you edit a Jupyter Notebook in VS Code or using the Jupyter Notebook interface, it follows the same process. First you add a cell. Next you define what programming language or kernel the notebook uses to process that cell. Finally, enter your content into the cell.

3. Click on the kernel name to choose whether the render is code or Markdown. Choose Markdown for the first cell. `# Test Sample  This is our first notebook.`
4. Add a second cell, this time make it a Python cell and add a print command. `print("Hello world!")`
5. Save your file as `test-notebook.ipynb`
6. Commit changes to your `build-sample` branch and publish it to the remote. **You don't need to merge your changes into `main`, but you can if you want.**

1.8 Build your Notebook binder with mybinder.org

After you publish a branch with a Jupyter notebook, now you can create a Binder to host it. MyBinder.org is a non-profit, free service that hosts your notebooks at a publicly accessible URL. 1. Open MyBinder.org 2. Enter your Github repo URL. It's the URL you can use to access your repo, https://github.com/lzdanski/test-sample, for example 3. Add the branch name for the branch you published with your notebook, **build-sample**. 4. **Launch** the Binder builder. When Binder finishes building your notebooks, it automatically opens a new browser window to access your notebook. You can now copy the URL to share your notebook, or download your notebook in different formats.

1.9 Binder tips and tricks

There are some quirks to using Binder - for example, if you make any changes to your Remote Branch, it breaks the dynamic hosting. You need to relaunch Binder to include any updates.

However, you can easily export your notebooks to other formats from Binder by going to **File»Save and export as** to download your Jupyter notebook as Markdown, LaTeX, or PDF.

Congratulations! You went through all the steps to create a new Github project in a public Github account and hosted a Jupyter notebook on Binder. Now you're ready to create your own writing sample.