

Testing Jupyter-based tools

Testing Jupyter-based tools

The proxied Jupyter tool is a useful place to develop code and analyses in a notebook style. Hub users can easily share their notebooks with other users by publishing notebooks as tools.

These instructions take you through *testing* the deployment of a Jupyter notebook based tool. Here, we'll assume that the short name for your tool is *toolname*. To test the notebook tool, it's handy to use the hub's [Workspace](#) tool, since this allows you to fully test the deployment in the context of the hub.

1. Create the tool

Once you have your notebook working to your satisfaction on the hub, you next create the tool to house it, and edit the invoke script. Once this is done, it is time to test.

NOTE that any new development, and any updates to existing tools, should make use of Debian10 containers. Develop these using the appropriate Jupyter tool and Workspace10.

2. Test invoke the tool

From the Workspace10 tool, or the Jupyter terminal tool, navigate to the directory where your tool's repo is located. For a tool you've called *toolname* and stored in the apps subdirectory of your home, this will be something like:

```
~/apps/toolname
```

Now, navigate to your tool's middleware directory, and call the invoke script for your tool, by typing:

```
cd middleware  
./invoke
```

Check the command line output to determine the success of your tool invoke call. Errors will display here if a problem is encountered. Use these to aid in your troubleshooting. If you see errors, you will need to revisit the tool sources, retesting to see if your fixes have worked, before going on with this procedure. Note that you may also see warnings displayed, as well as informational output. Neither warnings nor information indicate issues that need to be fixed.

A successful invoke script call will output in part:

The Jupyter notebook is running

...followed by a lengthy URL that reads, in part:

```
https://proxy.yourhub.org/weber/
```

Congratulations, your notebook-based tool is now running!

3. Check the running notebook

In the Workspace10 tool's command line output, note the informational output, denoted by lines starting in "I", and warning output, denoted by lines starting in "W", that is also displayed. These messages can be ignored safely; refer to the figure below.

Within the Workspace10 tool, you can now start a browser and paste into it the URL of the running Jupyter kernel. This test is not possible from outside your development environment, in order to protect your unreleased tool.

Copy the running kernel's URL

First, locate and highlight the kernel URL provided in the output from the invoke script.

Start the Workspace10 browser

Start the Firefox browser from the Workspace tool's menu. To do so, click the black button at the bottom left of the Workspace10 and access the Firefox menu item.

Navigate to the running kernel

A browser window will display, running inside your Workspace tool session on your hub. Finally, click (mouse wheel or both mouse buttons) to paste the running Jupyter kernel's URL into the browser navigation bar, as shown in this graphic

Now, the Workspace browser will display the running tool. This is the notebook running as a tool, and should be a good indication of how the tool will run once it is deployed.

One important error type you should watch for is the URL timeout. If there are URLs that your notebook needs access to in order to run, they will likely time out during this test. Collect any such URLs and include them in a support ticket on your hub to your hub administrators. The administrator will need to approve (whitelist) these URLs in order for them to be accessible to your notebook once it is a deployed tool. Be sure to explain this in your ticket, and clearly identify the tool name and the reason each URL is needed.

TESTING JUPYTER-BASED TOOLS

To stop testing, close the browser session running inside your Workspace10, then type "control-c" in the terminal where you called the invoke script. Once the prompt returns, your notebook kernel has stopped.

You can make any adjustments needed to the underlying code before you flag your tool as Uploaded and continue with the deployment process.