



Red Hat OpenShift Data Science 1

Working with notebooks

Create and collaborate on notebooks in your OpenShift Data Science notebook environment

Red Hat OpenShift Data Science 1 Working with notebooks

Create and collaborate on notebooks in your OpenShift Data Science notebook environment

Legal Notice

Copyright © 2021 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

Learn how to create and collaborate on notebooks in your OpenShift Data Science notebook environment.

Table of Contents

CHAPTER 1. CREATING AND IMPORTING NOTEBOOKS	3
1.1. CREATING A NEW NOTEBOOK	3
1.1.1. Notebook images for data scientists	3
1.2. IMPORTING AN EXISTING NOTEBOOK FILE FROM LOCAL STORAGE	4
1.3. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING JUPYTERLAB	4
1.4. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING THE COMMAND LINE INTERFACE	5
1.5. ADDITIONAL RESOURCES	6
CHAPTER 2. COLLABORATING ON NOTEBOOKS USING GIT	7
2.1. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING JUPYTERLAB	7
2.2. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING THE COMMAND LINE INTERFACE	7
2.3. UPDATING YOUR PROJECT WITH CHANGES FROM A REMOTE GIT REPOSITORY	8
2.4. PUSHING PROJECT CHANGES TO A GIT REPOSITORY	9
CHAPTER 3. WORKING WITH NOTEBOOKS ON OPENSIFT DATA SCIENCE	10
3.1. VIEWING PYTHON PACKAGES INSTALLED ON YOUR NOTEBOOK SERVER	10
3.2. INSTALLING PYTHON PACKAGES ON YOUR NOTEBOOK SERVER	11
3.3. UPDATING NOTEBOOK SERVER SETTINGS BY RESTARTING YOUR SERVER	12
CHAPTER 4. TROUBLESHOOTING COMMON PROBLEMS IN JUPYTERHUB	13
4.1. I SEE A 403: FORBIDDEN ERROR WHEN I LOG IN TO JUPYTERHUB	13
4.2. MY NOTEBOOK SERVER DOES NOT START	13
4.3. I SEE A DATABASE OR DISK IS FULL ERROR OR A NO SPACE LEFT ON DEVICE ERROR WHEN I RUN MY NOTEBOOK CELLS	14

CHAPTER 1. CREATING AND IMPORTING NOTEBOOKS

You can create a blank notebook or import a notebook from a number of different sources.

1.1. CREATING A NEW NOTEBOOK

You can create a new Jupyter notebook from an existing notebook container image to access its resources and properties. The JupyterHub Spawner contains a list of available container images that you can run as a single-user notebook server.

Prerequisites

- Ensure that you have logged in to Red Hat OpenShift Data Science.
- Ensure that you have logged in to JupyterHub and launched your notebook server.
- The notebook image exists in a registry, image stream, and is accessible.

Procedure

1. Click **File** → **New** → **Notebook**.
2. If prompted, select a kernel for your notebook from the list.
If you want to use a kernel, click **Select**. If you do not want to use a kernel, click **No Kernel**.

Verification

- Check that the notebook file is visible in JupyterHub.

1.1.1. Notebook images for data scientists

Red Hat OpenShift Data Science contains Jupyter notebook images optimized with industry-leading tools and libraries required for your data science work. To provide a consistent, stable platform for your model development, all notebook images contain the same version of Python. Notebook images available on Red Hat OpenShift Data Science are automatically upgraded to the latest supported version.

Red Hat OpenShift Data Science contains the following notebook images that are installed by default:

Table 1.1. Default notebook images

Image name	Description
Standard Data Science	Use the Standard Data Science notebook image for models that do not require TensorFlow or PyTorch.
TensorFlow	TensorFlow is an open source platform for machine learning. With TensorFlow, you can build, train and deploy your machine learning models. TensorFlow contains advanced data visualization features, such as computational graph visualizations. It also allows you to easily monitor and track the progress of your models.

Image name	Description
PyTorch	PyTorch is an open source machine learning library optimized for deep learning. If you are working with computer vision or natural language processing models, use the Pytorch notebook image.
Minimal Python	If you do not require advanced machine learning features, or additional resources for compute-intensive data science work, you can use the Minimal Python image to develop your models.


1.2. IMPORTING AN EXISTING NOTEBOOK FILE FROM LOCAL STORAGE

You can load an existing notebook from local storage into JupyterLab to continue work, or adapt a project for a new use case.

Prerequisites

- Credentials for logging in to JupyterHub.
- A launched and running notebook server.
- A notebook file exists in your local storage.

Procedure

1. In the **File Browser** in the left sidebar of the JupyterLab interface, click **Upload Files** ().
2. Locate and select the notebook file and click **Open**.
The file is displayed in the **File Browser**.

Verification

- The notebook file is displayed in the **File Browser** in the left sidebar of the JupyterLab interface.
- You can open the notebook file in JupyterLab.

1.3. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING JUPYTERLAB

You can use the JupyterLab user interface to clone a Git repository into your workspace to continue your work or integrate files from an external project.


Prerequisites

- A launched and running JupyterHub server.
- Read access for the Git repository you want to clone.

Procedure

1. Copy the HTTPS URL for the Git repository.
 - On GitHub, click **Code** → **HTTPS** and click the Clipboard button.
 - On GitLab, click **Clone** and click the Clipboard button under **Clone with HTTPS**.

2. In the JupyterLab interface click the **Git Clone** button ().

You can also click **Git** → **Clone a repository** in the menu, or click the Git icon () and click the **Clone a repository** button.

The *Clone a repo* dialog appears.

3. Enter the HTTPS URL of the repository that contains your notebook.
4. Click **CLONE**.
5. If prompted, enter your username and password for the Git repository.

Verification

- Check that the contents of the repository are visible in the file browser in JupyterLab, or run the **ls** command in the Terminal to verify that the repository is shown as a directory.

1.4. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING THE COMMAND LINE INTERFACE

You can use the command line interface to clone a Git repository into your workspace to continue your work or integrate files from an external project.

Prerequisites

- A launched and running JupyterHub server.

Procedure

1. Copy the HTTPS URL for the Git repository.
 - On GitHub, click **Code** → **HTTPS** and click the Clipboard button.
 - On GitLab, click **Clone** and click the Clipboard button under **Clone with HTTPS**.
2. In JupyterLab, click **File** → **New** → **Terminal** to open a Terminal window.
3. Enter the **git clone** command.

```
git clone git-clone-url
```

Replace *git-clone-url* with the HTTPS URL, for example:

```
[1234567890@jupyterhub-nb-jdoe ~]$ git clone https://github.com/example/myrepo.git
Cloning into myrepo...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
```

```
remote: Total 2821 (delta 1), reused 5 (delta 1), pack-reused 2810
Receiving objects: 100% (2821/2821), 39.17 MiB | 23.89 MiB/s, done.
Resolving deltas: 100% (1416/1416), done.
```

Verification

- Check that the contents of the repository are visible in the file browser in JupyterLab, or run the `ls` command in the terminal to verify that the repository is shown as a directory.

1.5. ADDITIONAL RESOURCES

- [Collaborating on notebooks using Git](#)

CHAPTER 2. COLLABORATING ON NOTEBOOKS USING GIT

If your notebooks or other files are stored in Git version control, you can import them from a Git repository onto your notebook server to work with them in JupyterHub. When you are ready, you can push your changes back to the Git repository so that others can review or use your models.

2.1. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING JUPYTERLAB

You can use the JupyterLab user interface to clone a Git repository into your workspace to continue your work or integrate files from an external project.


Prerequisites

- A launched and running JupyterHub server.
- Read access for the Git repository you want to clone.

Procedure

1. Copy the HTTPS URL for the Git repository.
 - On GitHub, click **Code** → **HTTPS** and click the Clipboard button.
 - On GitLab, click **Clone** and click the Clipboard button under **Clone with HTTPS**.

2. In the JupyterLab interface click the **Git Clone** button ().

You can also click **Git** → **Clone a repository** in the menu, or click the Git icon () and click the **Clone a repository** button.

The *Clone a repo* dialog appears.

3. Enter the HTTPS URL of the repository that contains your notebook.
4. Click **CLONE**.
5. If prompted, enter your username and password for the Git repository.

Verification

- Check that the contents of the repository are visible in the file browser in JupyterLab, or run the **ls** command in the Terminal to verify that the repository is shown as a directory.

2.2. IMPORTING AN EXISTING NOTEBOOK FILE FROM A GIT REPOSITORY USING THE COMMAND LINE INTERFACE

You can use the command line interface to clone a Git repository into your workspace to continue your work or integrate files from an external project.

Prerequisites

- A launched and running JupyterHub server.

Procedure

1. Copy the HTTPS URL for the Git repository.
 - On GitHub, click **Code** → **HTTPS** and click the Clipboard button.
 - On GitLab, click **Clone** and click the Clipboard button under **Clone with HTTPS**.
2. In JupyterLab, click **File** → **New** → **Terminal** to open a Terminal window.
3. Enter the **git clone** command.

```
git clone git-clone-url
```

Replace *git-clone-url* with the HTTPS URL, for example:

```
[1234567890@jupyterhub-nb-jdoe ~]$ git clone https://github.com/example/myrepo.git
Cloning into myrepo...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 2821 (delta 1), reused 5 (delta 1), pack-reused 2810
Receiving objects: 100% (2821/2821), 39.17 MiB | 23.89 MiB/s, done.
Resolving deltas: 100% (1416/1416), done.
```

Verification

- Check that the contents of the repository are visible in the file browser in JupyterLab, or run the **ls** command in the terminal to verify that the repository is shown as a directory.



2.3. UPDATING YOUR PROJECT WITH CHANGES FROM A REMOTE GIT REPOSITORY

You can pull changes made by other users into your data science project from a remote Git repository.

Prerequisites

- You have configured the remote Git repository.
- You have already imported the Git repository into JupyterLab, and the contents of the repository are visible in the file browser in JupyterLab.
- You have permissions to pull files from the remote Git repository to your local repository.
- You have credentials for logging in to JupyterHub.
- You have a launched and running JupyterHub server.

Procedure

1. In the JupyterLab interface, click the **Git** button ().
2. Click the **Pull latest changes** button ().

Verification

- You can view the changes pulled from the remote repository in the **History** tab of the Git pane.


2.4. PUSHING PROJECT CHANGES TO A GIT REPOSITORY

To build and deploy your application in a production environment, upload your work to a remote Git repository.

Prerequisites

- You have opened a notebook in the JupyterHub interface.
- You have already added the relevant Git repository to your notebook server.
- You have permission to push changes to the relevant Git repository.
- You have installed the Git version control extension.

Procedure

1. Click **File** → **Save All** to save any unsaved changes.
2. Click the Git icon () to open the Git pane in the JupyterHub interface.
3. Confirm that your changed files appear under **Changed**.
If your changed files appear under **Untracked**, click **Git** → **Simple Staging** to enable a simplified Git process.
4. Commit your changes.
 - a. Ensure that all files under **Changed** have a blue checkmark beside them.
 - b. In the **Summary** field, enter a brief description of the changes you made.
 - c. Click **Commit**.
5. Click **Git** → **Push to Remote** to push your changes to the remote repository.
6. When prompted, enter your Git credentials and click **OK**.

Verification

- Your most recently pushed changes are visible in the remote Git repository.

CHAPTER 3. WORKING WITH NOTEBOOKS ON OPENSIFT DATA SCIENCE

As a data scientist, you control what goes into your notebook server environment. You can install software as needed to ensure your server has everything your notebooks and your models require.



IMPORTANT

OpenShift Data Science sends notifications to nominated email addresses (usually your administrator) when the storage for your notebook server is 90% full, and again when it is completely full.

If you download a very large data set, your storage can fill up before your administrator receives a notification, so you might run out of room before your administrator can give you more storage.

To avoid this issue, Red Hat recommends streaming larger data sets from external services where possible. Alternatively, you can proactively request more storage from your administrator when you plan to work with a very large data set, to ensure you have sufficient space.

3.1. VIEWING PYTHON PACKAGES INSTALLED ON YOUR NOTEBOOK SERVER

You can check which Python packages are installed on your notebook server and which version of the package you have by running the **pip** tool in a notebook cell.

Prerequisites

- Log in to JupyterHub and open a notebook.

Procedure

1. Enter the following in a new cell in your notebook:

```
!pip list
```

2. Run the cell.

Verification

- The output shows an alphabetical list of all installed Python packages and their versions. For example, if you use this command immediately after creating a notebook server using the **Minimal** image the first packages shown are similar to the following:

Package	Version

aiohttp	3.7.3
alembic	1.5.2
appdirs	1.4.4
argo-workflows	3.6.1
argon2-cffi	20.1.0
async-generator	1.10

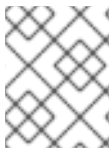
async-timeout	3.0.1
attrdict	2.0.1
attrs	20.3.0
backcall	0.2.0

Additional resources

- [Installing Python packages on your notebook server](#)

3.2. INSTALLING PYTHON PACKAGES ON YOUR NOTEBOOK SERVER

You can install Python packages that are not part of the default notebook server image by adding the package and the version to a **requirements.txt** file and then running the **pip install** command in a notebook cell.



NOTE

You can also install packages directly, but Red Hat recommends using a **requirements.txt** file so that it is easier to deploy your model later.

Prerequisites

- Log in to JupyterHub and open a notebook.

Procedure

1. Create a new text file using one of the following methods:
 - Click + to open a new launcher and click **Text file**.
 - Click **File** → **New** → **Text File**.
2. Rename the text file to **requirements.txt**.
 - a. Right-click on the name of the file and click **Rename Text**. The **Rename File** dialog opens.
 - b. Enter **requirements.txt** in the **New Name** field and click **Rename**.
3. Add the packages to install to the **requirements.txt** file.

```
altair
```

You can specify the exact version to install by using the **==** (equal to) operator, for example:

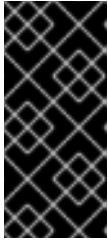
```
altair==4.1.0
```

To install multiple packages at the same time, place each package on a separate line.

4. Install the packages in **requirements.txt** to your server using a notebook cell.
 - a. Create a new cell in your notebook and enter the following command.

```
!pip install -r requirements.txt
```

- b. Run the cell by pressing Shift and Enter.



IMPORTANT

This installs the package on your notebook server, but you must still run the **import** directive in a code cell to use the package in your code.

```
import altair
```

Verification

- Confirm that the packages in **requirements.txt** appear in the list of packages installed on the notebook server. See [Viewing Python packages installed on your notebook server](#) for details.

3.3. UPDATING NOTEBOOK SERVER SETTINGS BY RESTARTING YOUR SERVER

You can update the settings on your notebook server by stopping and relaunching the notebook server. For example, if your server runs out of memory, you can restart the server to make the container size larger.

Prerequisites

- A running notebook server.
- Log in to JupyterHub.

Procedure

1. Click **File** → **Hub Control Panel**
The control panel opens in a new tab.
2. Click the **Stop my server** button.
This button disappears when the server stops.
3. Click **My Server** to restart the server and select new settings.

Verification

- The notebook server launcher opens when the server restarts.

Additional resources

- [Launching JupyterHub and starting a notebook server](#)

CHAPTER 4. TROUBLESHOOTING COMMON PROBLEMS IN JUPYTERHUB

If you are seeing errors in Red Hat OpenShift Data Science related to JupyterHub, your notebooks, or your notebook server, read this section to understand what could be causing the problem.

If you cannot see your problem here or in the release notes, contact Red Hat Support.

4.1. I SEE A 403: FORBIDDEN ERROR WHEN I LOG IN TO JUPYTERHUB

Problem

Your user name might not be added to the default user group or the default administrator group for OpenShift Data Science. Contact your administrator so that they can add you to the correct group/s.

Diagnosis

Check whether the user is part of either the default user group or the default administrator group.

1. Find the names of groups allowed access to JupyterHub.
 - a. Log in to OpenShift Dedicated web console.
 - b. Click **Workloads** → **ConfigMaps** and click on the **rhods-groups-config** ConfigMap to open it.
 - c. Click on the YAML tab and check the values for **admin_groups** and **allowed_groups**. These are the names of groups that have access to JupyterHub.

```
data:
  admin_groups: rhods-admins
  allowed_groups: rhods-users
```

2. Click **User management** → **Groups** and click on the name of each group to see its members.

Resolution

- If the user is not added to any of the groups allowed access to JupyterHub, follow [Adding users for OpenShift Data Science](#) to add them.
- If the user is already added to a group that is allowed to access JupyterHub, contact Red Hat Support.

4.2. MY NOTEBOOK SERVER DOES NOT START

Problem

The OpenShift Dedicated cluster that hosts your notebook server might not have access to enough resources, or the JupyterHub pod may have failed. Contact your administrator so that they can perform further checks.

Diagnosis

1. Log in to OpenShift Dedicated web console.

2. Delete and restart the notebook server pod for this user.
 - a. Click **Workloads** → **Pods** and set the **Project** to **rhods-notebooks**.
 - b. Search for the notebook server pod that belongs to this user exists, for example, **jupyterhub-nb-username-***.
If the notebook server pod exists, an intermittent failure may have occurred in the notebook server pod.

If the notebook server pod for the user does not exist, continue with diagnosis.
3. Check the resources currently available in the OpenShift Dedicated cluster against the resources required by the selected notebook server image.
If worker nodes with sufficient CPU and RAM are available for scheduling in the cluster, continue with diagnosis.
4. Check the state of the JupyterHub pod.

Resolution

- If there was an intermittent failure of the notebook server pod:
 - a. Delete the notebook server pod that belongs to the user.
 - b. Ask the user to start their notebook server again.
- If the notebook server does not have sufficient resources to run the selected notebook server image, either add more resources to the OpenShift Dedicated cluster, or choose a smaller image size.
- If the JupyterHub pod is in a **FAILED** state:
 - a. Retrieve the logs for the **jupyterhub-*** pod and send them to Red Hat Support for further evaluation.
 - b. Delete the **jupyterhub-*** pod.



WARNING

Ensure that you delete the correct pod. Do not delete the **jupyterhub-db-*** pod by mistake.

- If none of the previous resolutions apply, contact Red Hat Support.

4.3. I SEE A DATABASE OR DISK IS FULL ERROR OR A NO SPACE LEFT ON DEVICE ERROR WHEN I RUN MY NOTEBOOK CELLS

Problem

You might have run out of storage space on your notebook server. Contact your administrator so that they can perform further checks.

Diagnosis

1. Log in to JupyterHub and start the notebook server that belongs to the user having problems. If the notebook server does not start,
2. Check whether the user has run out of storage space.
 - a. Log in to OpenShift Dedicated web console.
 - b. Click **Workloads** → **Pods** and set the **Project** to **rhods-notebooks**.
 - c. Click the notebook server pod that belongs to this user, for example, **jupyterhub-nb-username-***.
 - d. Click **Logs**. The user has exceeded their available capacity if you see lines similar to the following:

```
Unexpected error while saving file: XXXX database or disk is full
```

Resolution

- Increase the user's available storage by expanding their persistent volume: [Expanding persistent volumes](#)
- Work with the user to identify files that can be deleted from the **/opt/app-root/src** directory to free up their existing storage space.