jupyterlab-slurm Documentation

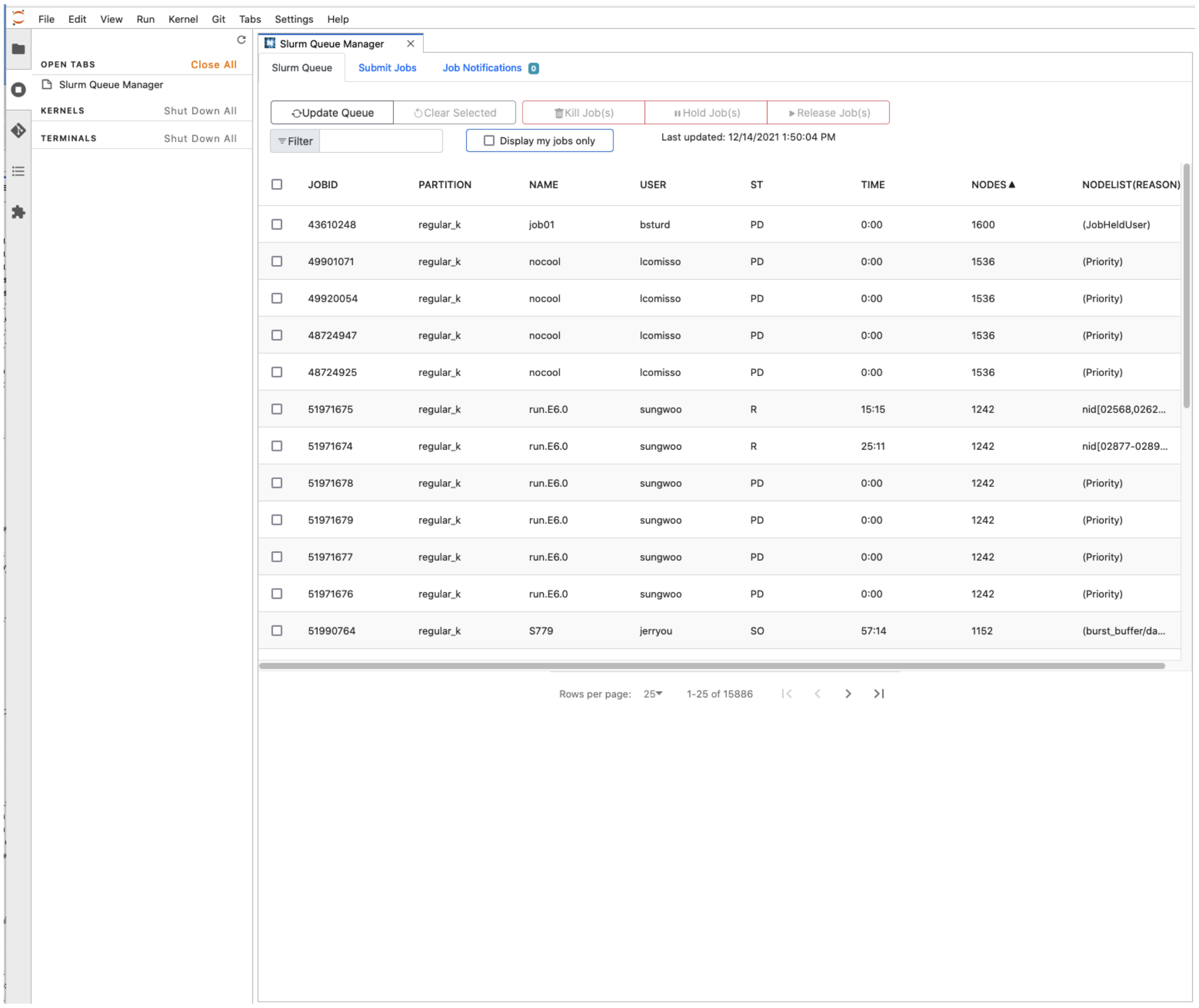
Release 0.1.3

Jon Hays, William Krinsman, NERSC

Nov 08, 2022

Contents:

`jupyterlab-slurm` is an extension for JupyterLab that interfaces with the Slurm Workload Manager, providing simple and intuitive controls for viewing and managing jobs on the queue, as well as submitting new jobs to the queue.



Last updated: 12/14/2021 1:50:04 PM

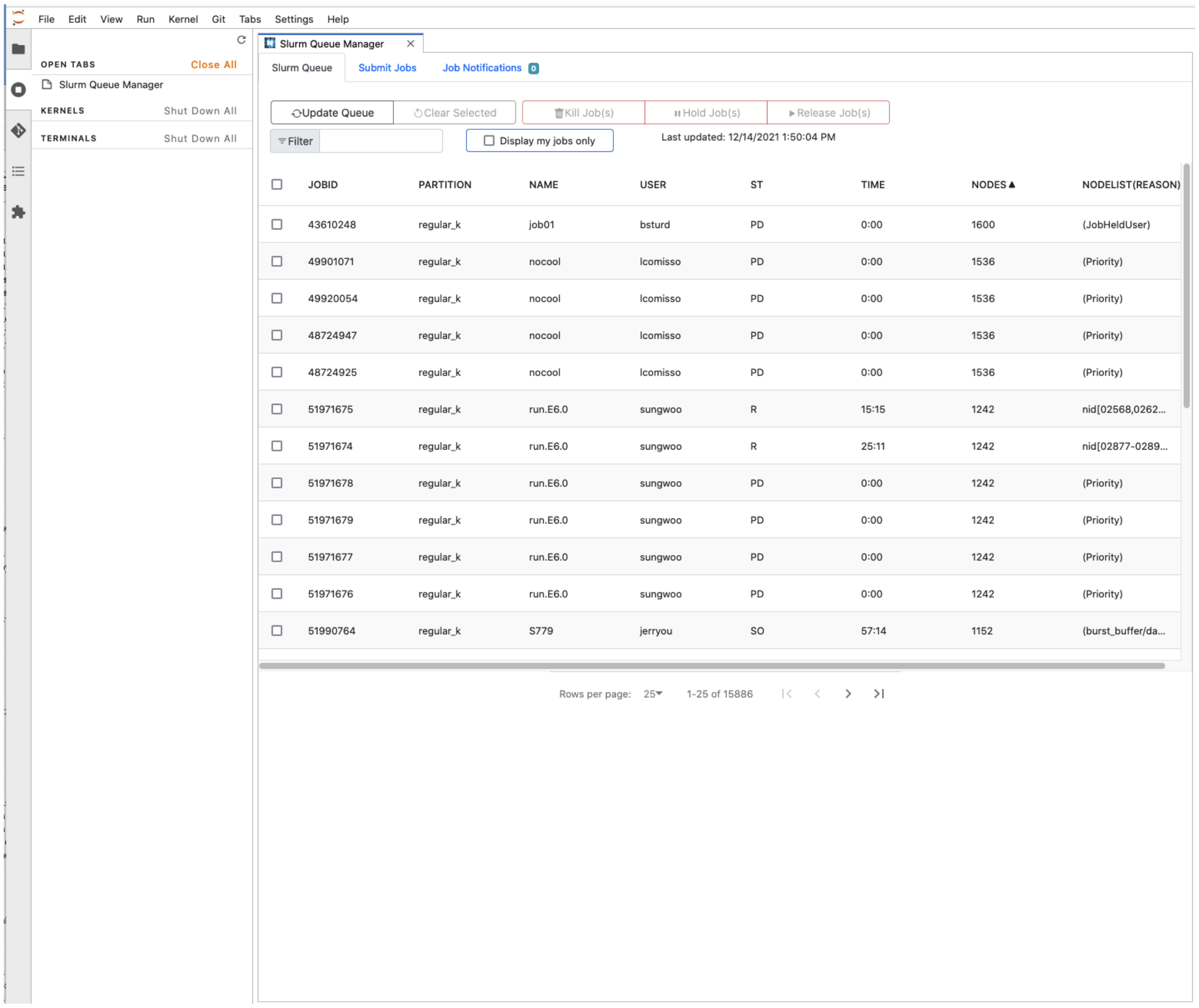
JOBID	PARTITION	NAME	USER	ST	TIME	NODES▲	NODELIST(REASON)
43610248	regular_k	job01	bsturd	PD	0:00	1600	(JobHeldUser)
49901071	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
49920054	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
48724947	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
48724925	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
51971675	regular_k	run.E6.0	sungwoo	R	15:15	1242	nid[02568,0262...
51971674	regular_k	run.E6.0	sungwoo	R	25:11	1242	nid[02877-0289...
51971678	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
51971679	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
51971677	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
51971676	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
51990764	regular_k	S779	jerryou	SO	57:14	1152	(burst_buffer/da...

Rows per page: 25 1-25 of 15886

CHAPTER 1

Overview

The `jupyterlab-slurm` extension provides a graphical user interface for interacting with the Slurm Workload Manager from within a JupyterLab session. The main focal point of the GUI is the Slurm queue, organized in an interactive table. Each row of the table corresponds to a pending job in the queue, while each column represents a particular field of metadata associated with each job (e.g, job ID, number of nodes allocated, job status, etc.).



	JOBID	PARTITION	NAME	USER	ST	TIME	NODES ▲	NODELIST(REASON)
☐	43610248	regular_k	job01	bsturd	PD	0:00	1600	(JobHeldUser)
☐	49901071	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
☐	49920054	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
☐	48724947	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
☐	48724925	regular_k	nocool	lcomisso	PD	0:00	1536	(Priority)
☐	51971675	regular_k	run.E6.0	sungwoo	R	15:15	1242	nid[02568,0262...
☐	51971674	regular_k	run.E6.0	sungwoo	R	25:11	1242	nid[02877-0289...
☐	51971678	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
☐	51971679	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
☐	51971677	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
☐	51971676	regular_k	run.E6.0	sungwoo	PD	0:00	1242	(Priority)
☐	51990764	regular_k	S779	jerryou	SO	57:14	1152	(burst_buffer/da...

The extension implements a few of the most common user operations in a Slurm system:

- Holding (or pausing) a job
- Releasing (or continuing) a job
- Killing a job
- Submitting new jobs

This extension was originally designed and built by student interns and staff at the National Energy Research Scientific Computing Center (NERSC), a high performance computing facility based out of Lawrence Berkeley National Laboratory (LBL). Our aim is to make the extension portable to any high performance computing system that uses Slurm.

CHAPTER 2

Installation

The jupyterlab-slurm extension requires both a client-side JupyterLab extension and a server-side Jupyter notebook server extension.

2.1 Prerequisites

Make sure you have the following prerequisites installed on your system before getting started.

- JupyterLab >= 0.35
- Node.js 5+
- Slurm

2.2 Install the notebook server extension from PyPi

```
pip install jupyterlab_slurm
```

If you are running Notebook 5.2 or earlier, enable the server extension by running

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

2.3 Install the JupyterLab extension from NPM

```
jupyter labextension install jupyterlab-slurm
```

Now, you should be all set!

CHAPTER 3

Usage

Proper usage of this extension requires prerequisite knowledge of using the Slurm Workload Manager, and writing batch scripts for job submission. For more information please visit the Slurm Documentation page, or seek further instruction through your institution's resources.

3.1 Getting Started

After installing the extension and launching JupyterLab, the extension can be found in the command palette under the name `Slurm Queue Manager`, and is listed under the `HPC TOOLS` section of the palette and the JupyterLab launcher. Open the extension through either of these two access points.

3.2 User view vs. global view

By default, the extension will be launched in "user view", meaning only jobs registered under your username in the system will be displayed in the queue. If you wish to view the entire queue, click the toggle labelled "Show my jobs only", and the queue will be switched to "global view". Note that the underlying Slurm command for retrieving queue data, `squeue`, is much more responsive for a smaller subset of jobs rather than the entire queue, so user view should be preferred unless you need to view other's jobs.

Note: Every field of the queue table is searchable via the `Search` entry box on the top right of the extension's GUI. You can sort the table based off any column as well. This means it is very simple to show only your active jobs, held jobs, etc.

3.3 Managing existing jobs in the queue

This extension provides an interface to three of the most common actions on existing jobs: `Kill Job(s)`, `Hold Job(s)`, and `Release Job(s)`. The underlying Slurm commands used are `scancel`, `scontrol hold`, and `scontrol release`, respectively. To carry out one of these actions on an existing job, simply select the row corresponding to the job, and click the appropriate button to submit the action. Multiple jobs can be selected by holding **Command/Ctrl** or **Shift** clicking, and then the same action can be requested on all selected jobs. Rows will become temporarily disabled until the request has finished. After a request completes, a manually dismissable alert will appear just beneath the queue, with background color corresponding to success (green) or failure (red).

3.4 Submitting new batch jobs

The Slurm extension also allows users to submit new jobs to the queue. The interface for doing so is accessed by clicking the `Submit Job` button. This button will launch a form that provides two different methods of job submission. After the job submission request completes, a success/failure alert will be displayed.

3.4.1 Submitting a job via path to existing batch script

The first field of the job submission form requires an absolute or relative path to an existing file that contains a valid batch script. The relative path is easy to acquire via the JupyterLab file browser, by right-clicking the desired file and selecting `Copy Path`.

3.4.2 Submitting a job via raw batch script

The second field of the job submission form will take in a raw batch script. This field can be good for writing a one-off script, or for pasting in an existing batch script and changing parameters on the fly.

Note: Sometimes submitting a job can take a long time! An alert message will appear once the job submission request has completed. We plan to add more visual feedback to indicate that a job submission is pending (e.g, a spinner) in the near future.

CHAPTER 4

Configuration

The project repository contains a configuration file used for setting various parameters for the front end JupyterLab extension, `src/slurm-config/config.json`. The following extension parameters are currently configurable:

- `squeueURL`: The string appended to the `baseUrl` to create the endpoint for retrieving queue data from the server extension. The default value is `/squeue`.
- `autoReload`: If `true`, the queue data will be reloaded automatically at a set interval. The default is `false`.
- `autoReloadRate`: The set time interval (in ms) for which the table will be automatically reloaded. The default is value is `60000`.
- `queueCols`: The names of the queue table's columns that will be displayed in the header; must be length 8. Default values correspond to the output of `squeue`.
- `cutoff`: The number of characters allowed to be displayed in a table cell before being truncated with an ellipses. The full output can be viewed through a tooltip when hovering over a truncated cell. The default value is `16` characters.
- `wordbreak`: If `true`, the content of a cell will be truncated at a word boundary. the default value is `true`.
- `escapeHtml`: If `true`, the following characters are escaped for truncated cell data: `<`, `>`, `&`, and `"`. The default value is `true`.

More configuration options will likely be added in the future. To setup the extension with configuration parameters other than the defaults listed above, you will need to navigate to the project repository and follow the directions for a development install of the JupyterLab extension. The server extension can still be installed directly from PyPi using `pip`, since it is not configurable at this point.

CHAPTER 5

Development

If you want to setup a development installation of the extension to configure/customize for your own use, or if you would like to contribute to further development of the extension, head over to the jupyterlab-slurm repository. There you can follow the directions for a development install, and submit issues/pull requests.

CHAPTER 6

Relevant Links

- GitHub Repository
- NPM Listing for Lab Extension
- PyPi Listing for Server Extension
- JupyterLab Documentation
- Slurm Documentation
- NERSC Homepage