`galyleo`: A shell utility to help you launch Jupyter notebooks in a simple, secure way

Marty Kandes, Ph.D.

Computational & Data Science Research Specialist
High-Performance Computing User Services Group
San Diego Supercomputer Center
University of California, San Diego

TSCC Workshop
Thursday, September 2nd, 2021
2:30PM - 3:00PM PDT

Today's Session

- What is `galyleo`? How does it work?
- How do you get started with `galyleo`?
- What are the limitations of `galyleo`?
- Q&A
- Hands-on Session

What is `galyleo`? How does it work?

`galyleo` is a shell utility to help you launch Jupyter notebooks on high-performance computing (HPC) systems – like TSCC – in a simple, secure way.

It works with the Satellite reverse proxy service and a batch job scheduler like PBS/Torqueto provide each Jupyter notebook server you start with its own one-time, token-authenticated HTTPS connection between the compute resources of the HPC system the notebook server is running on and your web browser.

`https://github.com/mkandes/galyleo`

History & Design of `galyleo`

`galyleo` is a 2nd-generation shell utility developed to orchestrate a user's interaction with both Satellite and PBS/Torque to start a Jupyter session within a batch job.

Developed to replace the 1st-generation `start-jupyter` shell utility; to support the Expanse User Portal and the SDSC HPC User Services Group long-term

Key features in design:

- Recreate same interactions with Satellite service
- Increase flexibility for users to configure software environment; but also try to make it simpler for them to do themselves
- Batch job script is generated completely on-the-fly
- Command-line argument driven

Getting Started with `galyleo` in 3 Easy Steps

1. Prepend its installation location to your `PATH`.

```
export PATH="/projects/builder-group/galyleo:${PATH}"
```

2. `launch` your Jupyter notebook session.

```
galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00
  --env-modules python,scipy/3.6 --jupyter notebook
```

3. Copy and paste the HTTPS URL into your web browser.

`galyleo launch` command-line options

```
galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00 \
  --env-modules python,scipy/3.6 --jupyter notebook
```

A number of command-line options allow you to configure:

- the compute resources required to run your Jupyter notebook session;
- the type of Jupyter interface you want to use for the session and the location of the notebook working directory; and
- the software environment that contains the `jupyter` notebook server and the other software packages you want to work with during the session.

Please see documentation for complete list of command-line options:

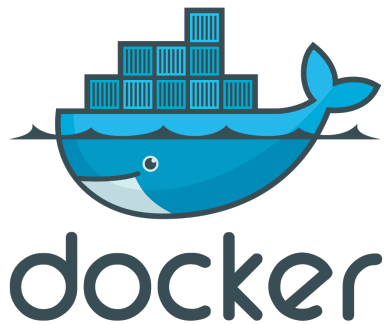
`https://github.com/mkandes/galyleo`

Defining your software environment

`--sif`

`--env-modules`

`--conda-env`



Limitations

- `galyleo` is still a prototype under active development; expect things to change over time without warning
- `galyleo` currently only works if you use `/bin/bash` for your `SHELL`; support for other `SHELL`s will likely be supported in the future
- `galyleo` has only limited support for PBS/Torque; initial design was for Slurm; TSCC will likely transition to Slurm in the not-too-distant future

Please report issues on GitHub:

`https://github.com/mkandes/galyleo`

Questions?

TSCC Quick Reference Guide

```
export PATH="/projects/builder-group/galyleo:${PATH}"

galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00 \
  --env-modules python,scipy/3.6 --jupyter notebook

galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00 \
  --sif r-notebook_latest.sif

export SIF_DIR='/projects/builder-group/singularity'
galyleo launch --account abc123 --partition gpu-hotel --cpus 3 \
  --time-limit 00:30:00  \
  --sif "${SIF_DIR}/pytorch/pytorch-v1.4.0-gpu-20200224.simg" \
  --bind '"/oasis,${TMPDIR}"' --nv

galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00 \
  --conda-env notebooks-sharing \
  --conda-init miniconda3/etc/profile.d/conda.sh

galyleo launch --account abc123 --cpus 1 --time-limit 00:30:00 \
  --conda-env notebooks-sharing \
  --conda-pack notebooks-sharing.tar.gz \
  --scratch-dir '"${TMPDIR}"'
```