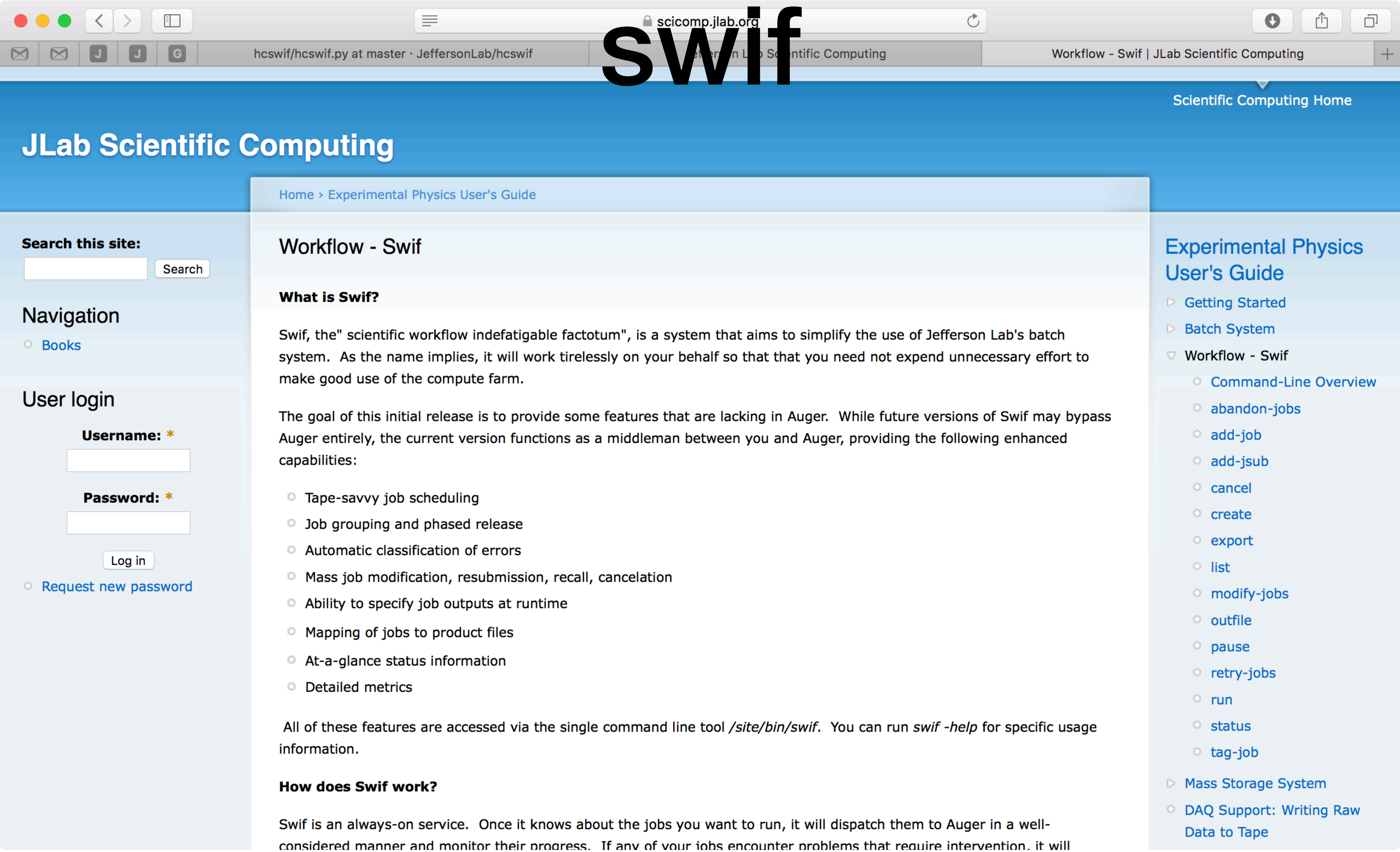
hcswif

Quick and easy SWIF job submission wrapper

John Matter
Joint Hall A & C Data Analysis Workshop
June 25, 2018

swif



Workflow - Swif

What is Swif?

Swif, the" scientific workflow indefatigable factotum", is a system that aims to simplify the use of Jefferson Lab's batch system. As the name implies, it will work tirelessly on your behalf so that that you need not expend unnecessary effort to make good use of the compute farm.

The goal of this initial release is to provide some features that are lacking in Auger. While future versions of Swif may bypass Auger entirely, the current version functions as a middleman between you and Auger, providing the following enhanced capabilities:

- Tape-savvy job scheduling
- Job grouping and phased release
- Automatic classification of errors
- Mass job modification, resubmission, recall, cancelation
- Ability to specify job outputs at runtime
- Mapping of jobs to product files
- At-a-glance status information
- Detailed metrics

All of these features are accessed via the single command line tool */site/bin/swif*. You can run *swif -help* for specific usage information.

How does Swif work?

Swif is an always-on service. Once it knows about the jobs you want to run, it will dispatch them to Auger in a well-considered manner and monitor their progress. If any of your jobs encounter problems that require intervention, it will

hcswif

- https://github.com/JeffersonLab/hcswif
- A Python3 script that makes workflow/job creation swifter!
- You give hcswif some info, and it writes a JSON file
- The JSON file, describing a swif workflow, can then be imported and run
- Primarily intended for replay of Hall C data, but can (kind of) help with other jobs

hcswif

- Example swif syntax:

```
swif create “MyWorkflow”
swif add-job “MyWorkflow” -project c-comm2017
                          -name myjob1
                          -input myfile1 /mss/hallc/spring17/raw/coin_all_02439.dat
                          -input myfile2 /mss/hallc/spring17/raw/coin_all_02440.dat
                          -cores 1 -ram 2G -time 4h
                          -stdout /volatile/hallc/blahblah/myjob1.out
                          -stderr /volatile/hallc/blahblah/myjob1.err
                          “/home/me/myscript.sh some args 123”
                          …

swif add-job “MyWorkflow” -project c-comm2017 -name myjob2
                          …

swif add-job “MyWorkflow” -project c-comm2017 -name myjob3
                          …

swif run MyWorkflow
```

- Cumbersome when analyzing large numbers of runs
- Can we make this easier?

hcswif

hcswif has two modes

replay

- Specific to hallc_replay
- Minimum input is **spectrometer** and **run number(s)**
- Guesses which replay script you want, but you can specify one explicitly

command

- Fairly general
- Minimum input is **command(s) to run**
- Environment variables can be tricky!

hcswif

hcswif has two modes

replay

- Specific to hallc_replay
- Minimum input is **spectrometer** and **run number(s)**
- Guesses which replay script you want, but you can specify one explicitly

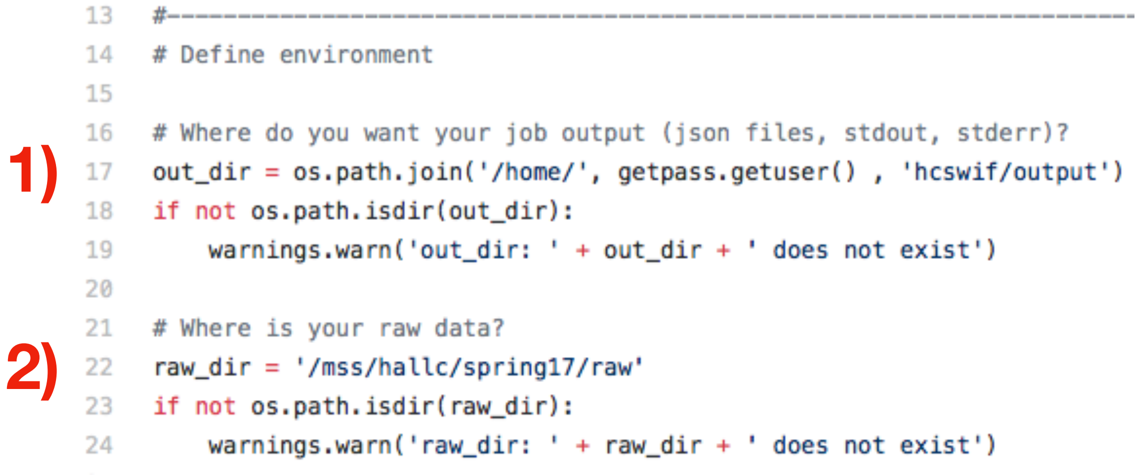
command

- Fairly general
- Minimum input is **command(s) to run**
- Environment variables can be tricky!

hcswif - setup

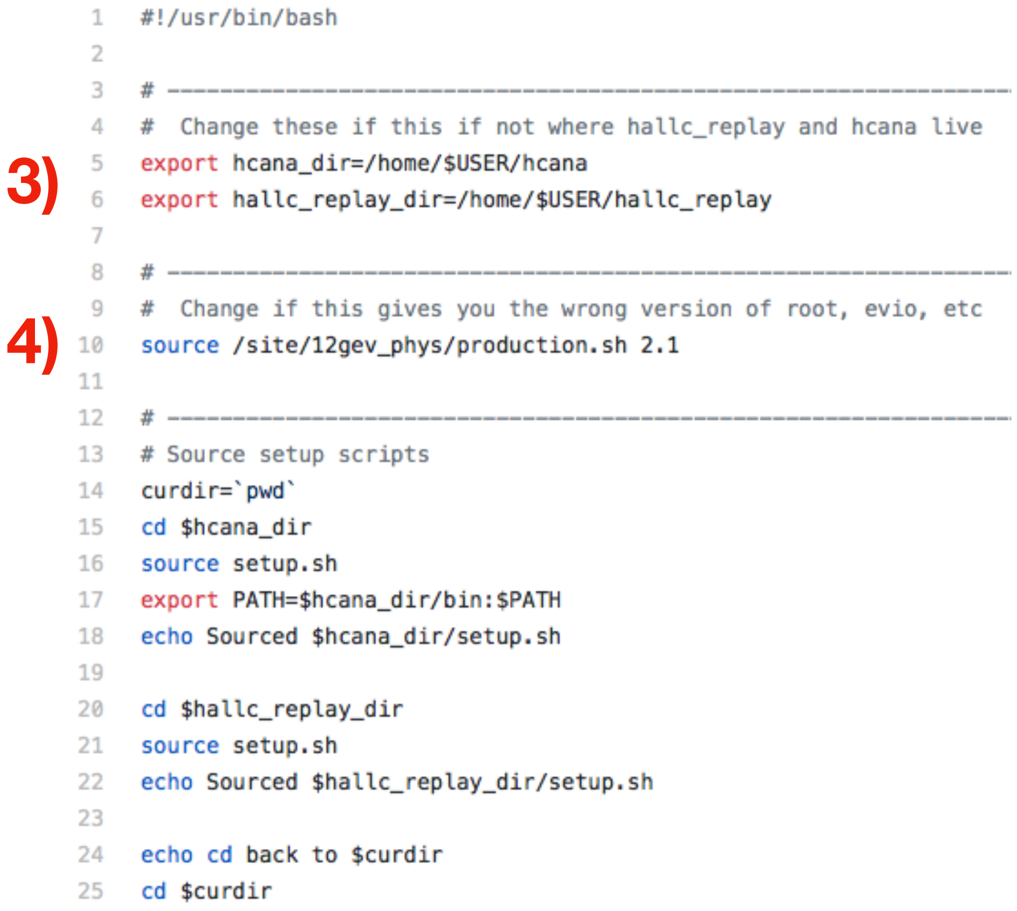
- git clone https://github.com/JeffersonLab/hcswif
- cd hcswif, edit hcswif.py and setup scripts

hcswif.py



```python
#------------------------------------------------------------------------------
# Define environment

# Where do you want your job output (json files, stdout, stderr)?
out_dir = os.path.join('/home/', getpass.getuser() , 'hcswif/output')
if not os.path.isdir(out_dir):
    warnings.warn('out_dir: ' + out_dir + ' does not exist')

# Where is your raw data?
raw_dir = '/mss/hallc/spring17/raw'
if not os.path.isdir(raw_dir):
    warnings.warn('raw_dir: ' + raw_dir + ' does not exist')
```

setup.sh



```bash
#!/usr/bin/bash

# ------------------------------------------------------------------------------
#  Change these if this if not where hallc_replay and hcana live
export hcana_dir=/home/$USER/hcana
export hallc_replay_dir=/home/$USER/hallc_replay

# ------------------------------------------------------------------------------
#  Change if this gives you the wrong version of root, evio, etc
source /site/12gev_phys/production.sh 2.1

# ------------------------------------------------------------------------------
# Source setup scripts
curdir=`pwd`
cd $hcana_dir
source setup.sh
export PATH=$hcana_dir/bin:$PATH
echo Sourced $hcana_dir/setup.sh

cd $hallc_replay_dir
source setup.sh
echo Sourced $hallc_replay_dir/setup.sh

echo cd back to $curdir
cd $curdir
```

hcswif - replay mode

- Suppose I want to replay SHMS events from several dozen coincidence runs

```
$ ./hcswif.py --mode replay --spectrometer SHMS_COIN --run 2187-2212 2023-2066
2049 --events 50000 --project c-comm2017 --name myswifjobs
Wrote: /some/directory/myswifjobs.json
```

- That’s it! 71 jobs with one command. Each run gets its own job.

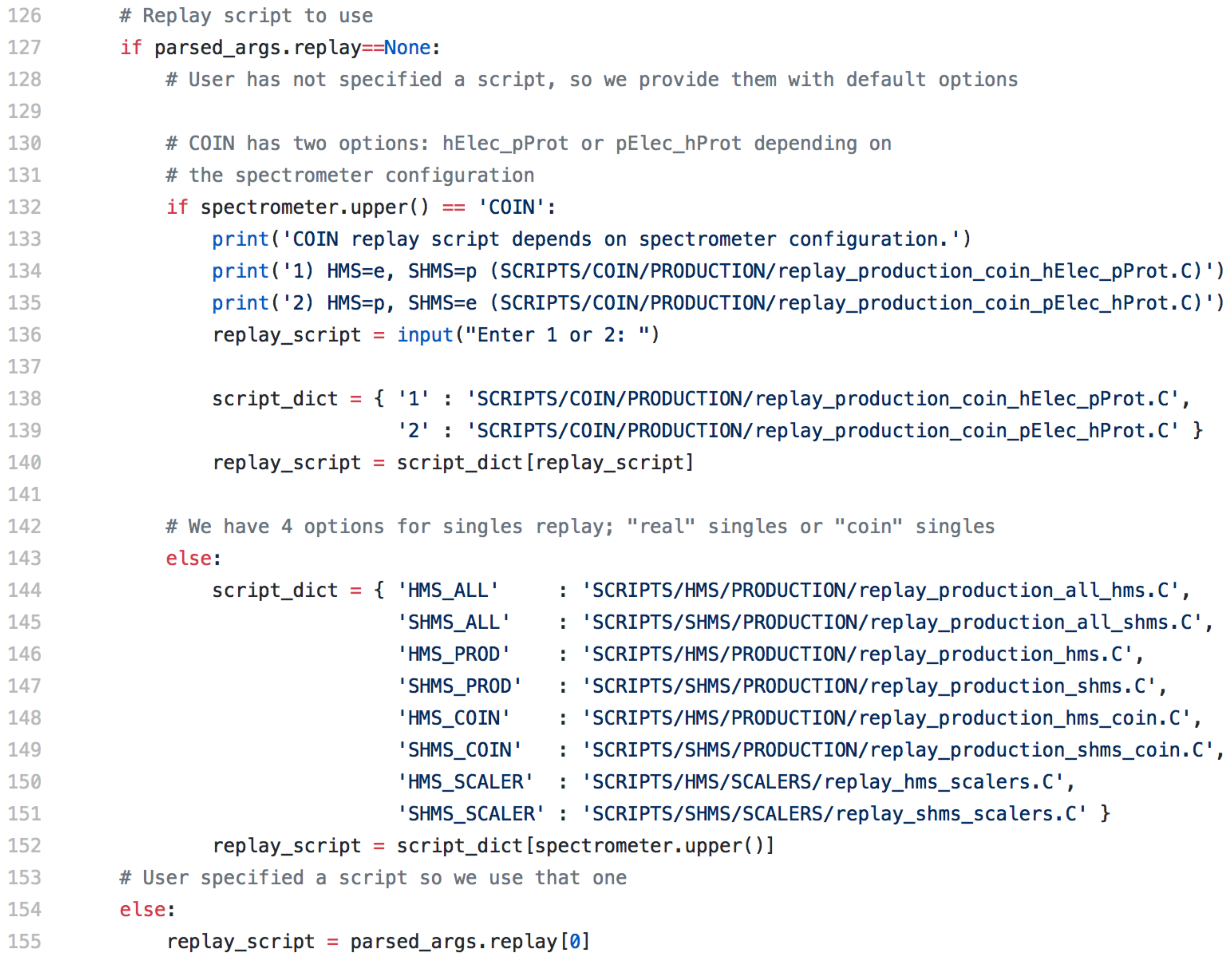
- Can also specify runs with a file (one run per line)

```
--run file list_of_runs.txt
```

- Guesses replay script based one `--spectrometer` argument, but can specify explicitly

```
--replay SCRIPTS/SHMS/SCALERS/replay_shms_scalers.C
```

hcswif - replay mode



```
# Replay script to use
if parsed_args.replay==None:
    # User has not specified a script, so we provide them with default options

    # COIN has two options: hElec_pProt or pElec_hProt depending on
    # the spectrometer configuration
    if spectrometer.upper() == 'COIN':
        print('COIN replay script depends on spectrometer configuration.')
        print('1) HMS=e, SHMS=p (SCRIPTS/COIN/PRODUCTION/replay_production_coin_hElec_pProt.C)')
        print('2) HMS=p, SHMS=e (SCRIPTS/COIN/PRODUCTION/replay_production_coin_pElec_hProt.C)')
        replay_script = input("Enter 1 or 2: ")

        script_dict = { '1' : 'SCRIPTS/COIN/PRODUCTION/replay_production_coin_hElec_pProt.C',
                        '2' : 'SCRIPTS/COIN/PRODUCTION/replay_production_coin_pElec_hProt.C' }
        replay_script = script_dict[replay_script]

    # We have 4 options for singles replay; "real" singles or "coin" singles
    else:
        script_dict = { 'HMS_ALL'     : 'SCRIPTS/HMS/PRODUCTION/replay_production_all_hms.C',
                        'SHMS_ALL'    : 'SCRIPTS/SHMS/PRODUCTION/replay_production_all_shms.C',
                        'HMS_PROD'    : 'SCRIPTS/HMS/PRODUCTION/replay_production_hms.C',
                        'SHMS_PROD'   : 'SCRIPTS/SHMS/PRODUCTION/replay_production_shms.C',
                        'HMS_COIN'    : 'SCRIPTS/HMS/PRODUCTION/replay_production_hms_coin.C',
                        'SHMS_COIN'   : 'SCRIPTS/SHMS/PRODUCTION/replay_production_shms_coin.C',
                        'HMS_SCALER'  : 'SCRIPTS/HMS/SCALERS/replay_hms_scalers.C',
                        'SHMS_SCALER' : 'SCRIPTS/SHMS/SCALERS/replay_shms_scalers.C' }
        replay_script = script_dict[spectrometer.upper()]
# User specified a script so we use that one
else:
    replay_script = parsed_args.replay[0]
```

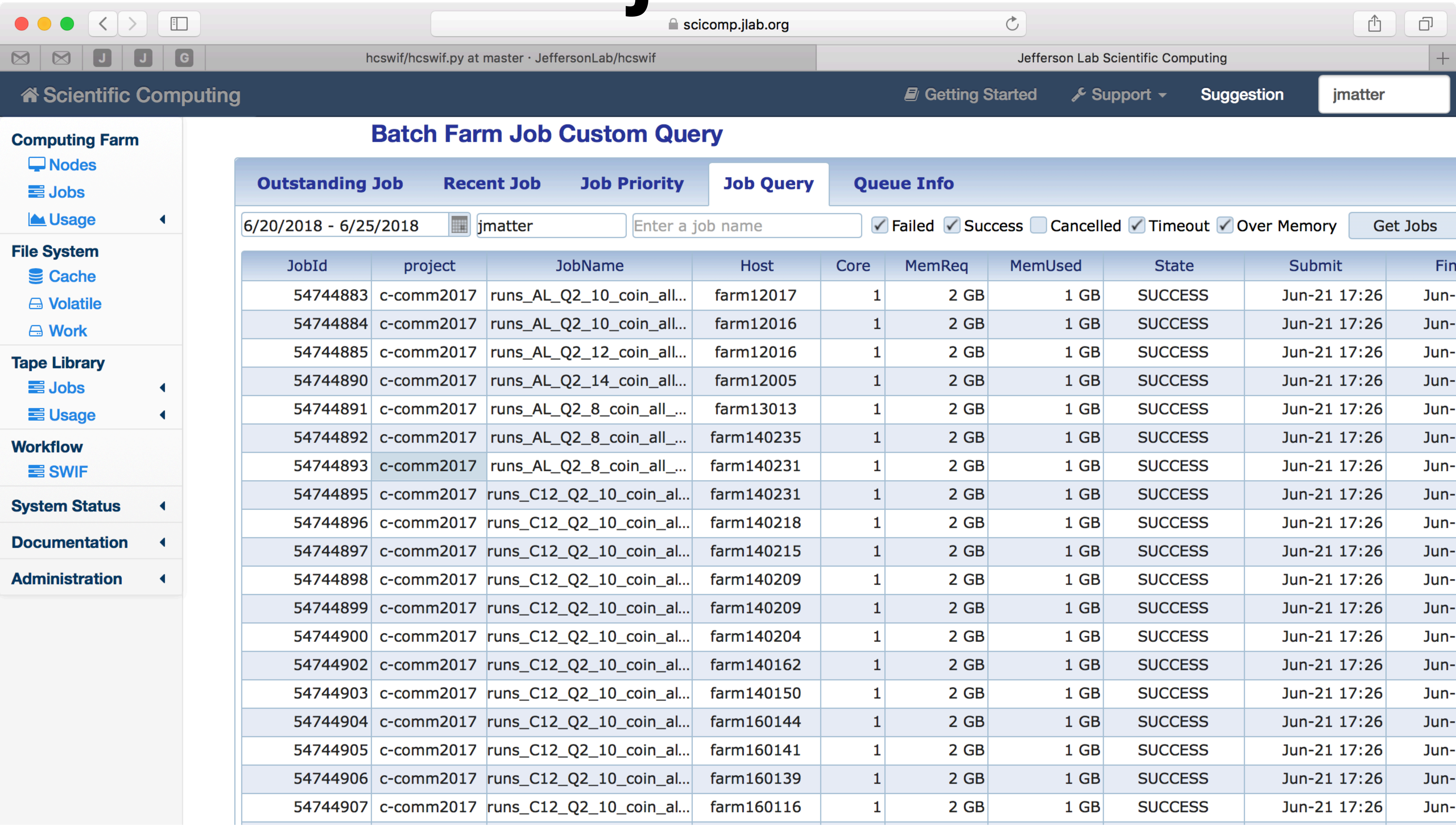
hcswif - JSON workflows

```
$ python -m json.tool myswifjob.json
{
    "jobs": [
        {
            "command": “/home/me/myscript.sh some args 123”,
            "cpuCores": 1,
            "diskBytes": 5000000000,
            "input": [
                {
                    "local": "coin_all_02049.dat",
                    "remote": "/mss/hallc/spring17/raw/coin_all_02049.dat"
                }
            ],
            "name": “myJob1",
            "os": "centos7",
            "project": "c-comm2017",
            "ramBytes": 8000000000,
            "shell": "/usr/bin/bash",
            "stderr": “/volatile/hallc/blahblah/myjob1.out”,
            . . .
```

swif - import and run

```
$ swif import -file output/myswifjobs.json
$ swif list
workflow_id                    = 45683
workflow_name                  = myswifjobs
workflow_user                  = jmatter
suspended                      = 1
jobs                           = 71
undispatched                   = 71
attempts                       = 0
create_ts                      = 2018-06-25 12:54:18.0
update_ts                      = 2018-06-25 12:56:49.0
current_ts                     = 2018-06-25 13:04:49.0
$ swif run myswifjobs
```

swif - job status



Batch Farm Job Custom Query

6/20/2018 - 6/25/2018 | jmatter | Enter a job name | ☑ Failed ☑ Success ☐ Cancelled ☑ Timeout ☑ Over Memory

JobId	project	JobName	Host	Core	MemReq	MemUsed	State	Submit	Fin
54744883	c-comm2017	runs_AL_Q2_10_coin_all...	farm12017	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744884	c-comm2017	runs_AL_Q2_10_coin_all...	farm12016	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744885	c-comm2017	runs_AL_Q2_12_coin_all...	farm12016	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744890	c-comm2017	runs_AL_Q2_14_coin_all...	farm12005	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744891	c-comm2017	runs_AL_Q2_8_coin_all_...	farm13013	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744892	c-comm2017	runs_AL_Q2_8_coin_all_...	farm140235	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744893	c-comm2017	runs_AL_Q2_8_coin_all_...	farm140231	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744895	c-comm2017	runs_C12_Q2_10_coin_al...	farm140231	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744896	c-comm2017	runs_C12_Q2_10_coin_al...	farm140218	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744897	c-comm2017	runs_C12_Q2_10_coin_al...	farm140215	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744898	c-comm2017	runs_C12_Q2_10_coin_al...	farm140209	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744899	c-comm2017	runs_C12_Q2_10_coin_al...	farm140209	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744900	c-comm2017	runs_C12_Q2_10_coin_al...	farm140204	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744902	c-comm2017	runs_C12_Q2_10_coin_al...	farm140162	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744903	c-comm2017	runs_C12_Q2_10_coin_al...	farm140150	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744904	c-comm2017	runs_C12_Q2_10_coin_al...	farm160144	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744905	c-comm2017	runs_C12_Q2_10_coin_al...	farm160141	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744906	c-comm2017	runs_C12_Q2_10_coin_al...	farm160139	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-
54744907	c-comm2017	runs_C12_Q2_10_coin_al...	farm160116	1	2 GB	1 GB	SUCCESS	Jun-21 17:26	Jun-

swif - job status

What if I get some kind of error?

- Does your file actually exist? Do you have hallc_replay set up correctly? Fix it and then do `swif retry-jobs`
- Not enough RAM or time??

```
swif modify-jobs myswifjobs -time add -1h -
ram add 1gb -names -regexp ‘.*’
```

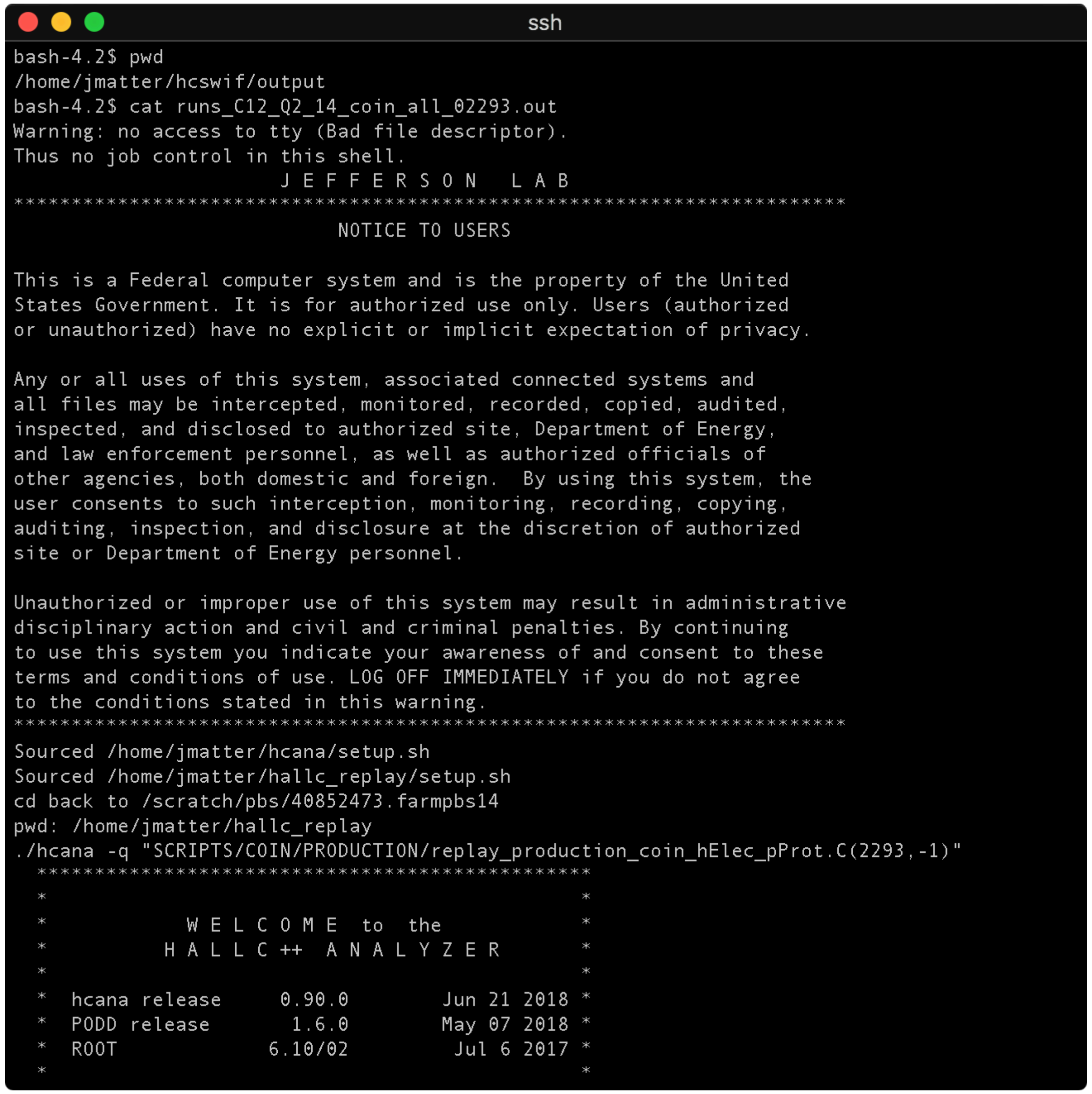
```
swif retry-jobs -workflow myswifjobs -name -
regexp ‘.*’
```

See https://scicomp.jlab.org/docs/swif for more details

hcswif - output

Each job generates a `workflow_jobname.out` and `workflow_jobname.err` log

Use these to diagnose problems



```
bash-4.2$ pwd
/home/jmatter/hcswif/output
bash-4.2$ cat runs_C12_Q2_14_coin_all_02293.out
Warning: no access to tty (Bad file descriptor).
Thus no job control in this shell.
                       J E F F E R S O N   L A B
************************************************************************
                            NOTICE TO USERS

This is a Federal computer system and is the property of the United
States Government. It is for authorized use only. Users (authorized
or unauthorized) have no explicit or implicit expectation of privacy.

Any or all uses of this system, associated connected systems and
all files may be intercepted, monitored, recorded, copied, audited,
inspected, and disclosed to authorized site, Department of Energy,
and law enforcement personnel, as well as authorized officials of
other agencies, both domestic and foreign.  By using this system, the
user consents to such interception, monitoring, recording, copying,
auditing, inspection, and disclosure at the discretion of authorized
site or Department of Energy personnel.

Unauthorized or improper use of this system may result in administrative
disciplinary action and civil and criminal penalties. By continuing
to use this system you indicate your awareness of and consent to these
terms and conditions of use. LOG OFF IMMEDIATELY if you do not agree
to the conditions stated in this warning.
************************************************************************
Sourced /home/jmatter/hcana/setup.sh
Sourced /home/jmatter/hallc_replay/setup.sh
cd back to /scratch/pbs/40852473.farmpbs14
pwd: /home/jmatter/hallc_replay
./hcana -q "SCRIPTS/COIN/PRODUCTION/replay_production_coin_hElec_pProt.C(2293,-1)"
  ************************************************
  *                                              *
  *            W E L C O M E  to  the            *
  *          H A L L C ++  A N A L Y Z E R       *
  *                                              *
  *  hcana release     0.90.0        Jun 21 2018 *
  *  PODD release       1.6.0        May 07 2018 *
  *  ROOT             6.10/02         Jul 6 2017 *
  *                                              *
```

hcswif

hcswif has two modes

replay

- Specific to hallc_replay
- Minimum input is **spectrometer** and **run number(s)**
- Guesses which replay script you want, but you can specify

command

- Fairly general
- Minimum input is **command(s) to run**
- Environment variables can be tricky!

hcswif - command mode

- Will currently only generate one job. (Sorry! I’ve been meaning to allow reading a file that lists one command per line, generating a job for each line.)
- Do you need to read files from tape? Two options for specifying files in command mode:
    1. Use explicit jget in your shell script
    2. Give hcswif a filelist (one filename /mss/blahblah/… per line)

hcswif - command mode

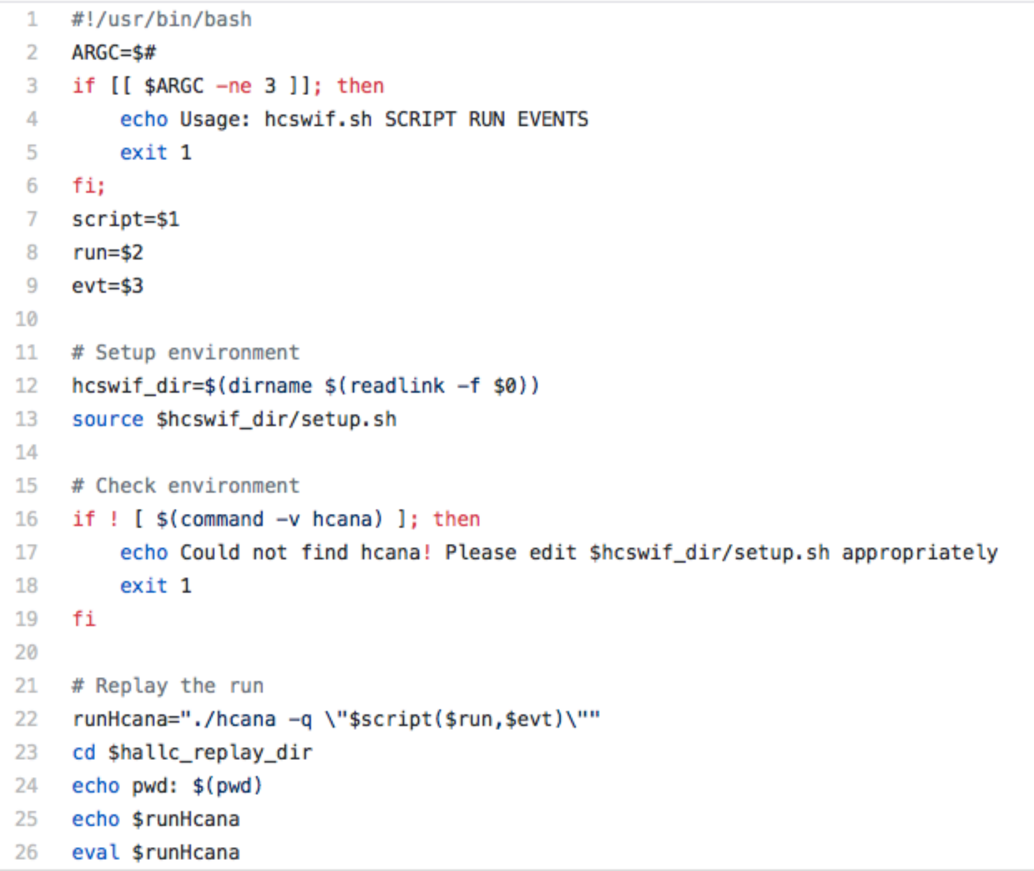
- For replay mode, environment variables are taken care of during the initial hcswif setup
- For command mode, you need to do this yourself
- swif jobs don’t use your .bashrc or .cshrc or even start in your home directory!
- You need to do *everything* inside the script that your job runs, including any sourcing of setup scripts

hcswif - command mode

- Let’s replay one coincidence run but this time using command mode
- Replay mode runs a bash script hcswif.sh (a wrapper for hcana)

```
./hcswif.sh SCRIPTS/HMS/PRODUCTION/replay_production_hms_coin.C 2049 -1
```

hcswif.sh



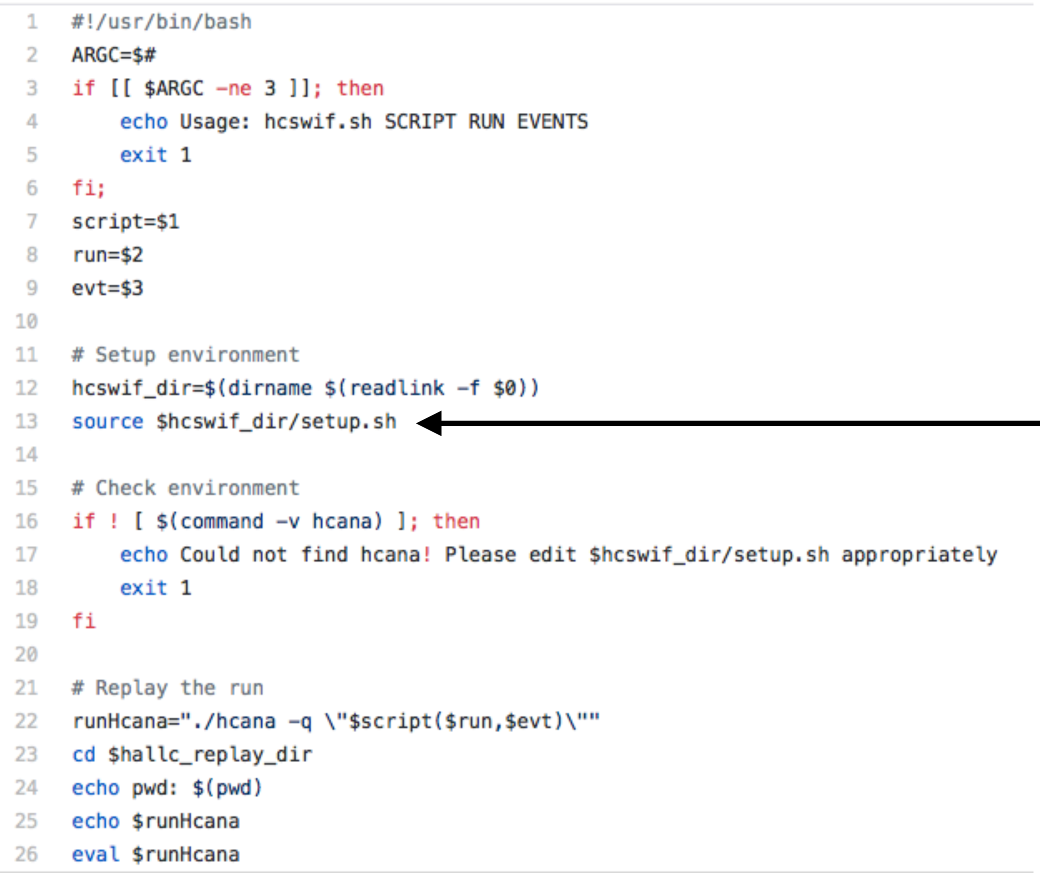
```
#!/usr/bin/bash
ARGC=$#
if [[ $ARGC -ne 3 ]]; then
    echo Usage: hcswif.sh SCRIPT RUN EVENTS
    exit 1
fi;
script=$1
run=$2
evt=$3

# Setup environment
hcswif_dir=$(dirname $(readlink -f $0))
source $hcswif_dir/setup.sh

# Check environment
if ! [ $(command -v hcana) ]; then
    echo Could not find hcana! Please edit $hcswif_dir/setup.sh appropriately
    exit 1
fi

# Replay the run
runHcana="./hcana -q \"$script($run,$evt)\""
cd $hallc_replay_dir
echo pwd: $(pwd)
echo $runHcana
eval $runHcana
```

hcswif - command mode

- Let’s replay one coincidence run but this time using command mode
- Replay mode runs a bash script hcswif.sh (a wrapper for hcana)

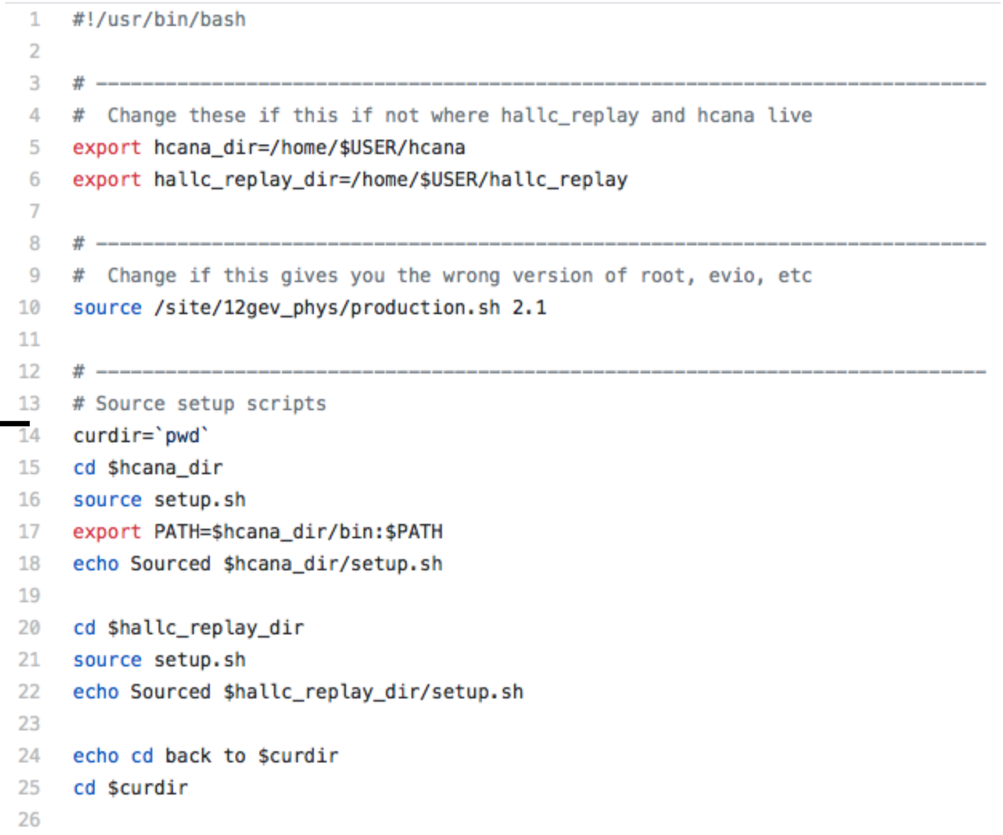
```
./hcswif.sh SCRIPTS/HMS/PRODUCTION/replay_production_hms_coin.C 2049 -1
```

hcswif.sh



```
#!/usr/bin/bash
ARGC=$#
if [[ $ARGC -ne 3 ]]; then
    echo Usage: hcswif.sh SCRIPT RUN EVENTS
    exit 1
fi;
script=$1
run=$2
evt=$3

# Setup environment
hcswif_dir=$(dirname $(readlink -f $0))
source $hcswif_dir/setup.sh

# Check environment
if ! [ $(command -v hcana) ]; then
    echo Could not find hcana! Please edit $hcswif_dir/setup.sh appropriately
    exit 1
fi

# Replay the run
runHcana="./hcana -q \"$script($run,$evt)\""
cd $hallc_replay_dir
echo pwd: $(pwd)
echo $runHcana
eval $runHcana
```

setup.sh



```
#!/usr/bin/bash

# -----------------------------------------------------------------------------
#  Change these if this if not where hallc_replay and hcana live
export hcana_dir=/home/$USER/hcana
export hallc_replay_dir=/home/$USER/hallc_replay

# -----------------------------------------------------------------------------
#  Change if this gives you the wrong version of root, evio, etc
source /site/12gev_phys/production.sh 2.1

# -----------------------------------------------------------------------------
# Source setup scripts
curdir=`pwd`
cd $hcana_dir
source setup.sh
export PATH=$hcana_dir/bin:$PATH
echo Sourced $hcana_dir/setup.sh

cd $hallc_replay_dir
source setup.sh
echo Sourced $hallc_replay_dir/setup.sh

echo cd back to $curdir
cd $curdir

```

hcswif - command mode

```
$ echo /mss/hallc/spring17/raw/coin_all_02049.dat > myswifjob2_files
```

```
$ ./hcswif.py --mode command --command “./hcswif.sh SCRIPTS/COIN/
PRODUCTION/replay_production_coin_hElec_pProt.C 2049 -1” --name
myswifjob2 --project c-comm2017 --filelist output/myswifjob2_files
Wrote: /some/directory/myswifjob2.json
```

```
$ swif import -file /some/directory/myswifjob2.json
$ swif run myswifjob2
```

hcswif - command mode

Very rough skeleton of a calibration script

```
#!/bin/bash
runNumber=$1
source /home/me/hcswif/setup.sh

# Replay calibration run
cd /home/me/hallc_replay
./hcana -q SCRIPTS/replay_uncalib.C($runNumber,-1)

# Get calibration from calibration run
root -q my_calib_macro.C($runNumber,-1)
mv CALIB/new_calib CALIB/old_calib

# Replay with new calibration
./hcana -q SCRIPTS/replay.C($runNumber,-1)

# Get some kind of diagnostic plots
./root -q CALIB/calib_plots.C($runNumber,-1)
```