ilifu Online Training

Jeremy Smith

User Training Workshop – Advanced Training #2 - Applications

15 May 2020

Support Team

- Jeremy Smith
  Operations Manager
- Jordan Collier
  Astronomy Support
- Dane Kennedy
  Bioinformatics Support
- Mike Currin
  ilifu Operations

Getting help

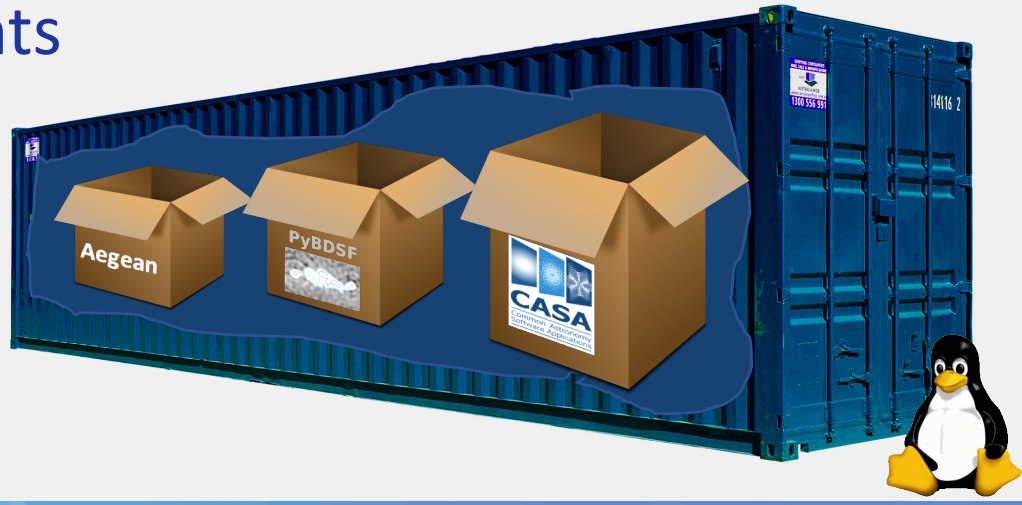
- Support contact
  support@ilifu.ac.za
- User documentation
  http://docs.ilifu.ac.za/#/
- ilifu System Status
  https://status.ilifu.ac.za/
- Training videos
  http://www.ilifu.ac.za/il/accessing-facilities/training

Topics – Advanced Training #2

- Singularity
  - Sourcing a container
  - Building a container
- Creating a custom kernel for JupyterHub
- Python virtualenv
- Using R containers and R-Studio
- Using Nextflow (Bioinformatics)
- CARTA
- SARAO Archive

Singularity containers

- Encapsulated software environments
- A software stack that contains everything required to run an application/workflow, including files, environmental variables, libraries and dependencies
- Containers accessible across platforms and services, allowing sharing of application environments



Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – sourcing a container

- Supported Containers

  available containers
- Project containers
  - /<group>/software/containers
- Containers from Docker

  singularity pull docker://<repo>/<image>

  OR

  singularity build <container_name>.simg docker://<repo>/<image>

  https://sylabs.io/guides/3.5/user-guide/singularity_and_docker.html

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – building a container

- Building a container requires root access
- Build a container on your personal computer and migrate to ilifu cluster storage
- Singularity is backwards compatible
- Current version: Singularity v3.5.2

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – building a container

- Build a container:
  - From a repo
    - docker://
    - shub://
  - From an existing container
  - From a Singularity definition file (recipe)

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
  - source: os or local image
- %files
  - copy files from base os into the container
- %environment
  - set environment variable for when containeri s
- %post
  - installation steps for software/libraries, etc

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- **Header**
- %files
- %environment
- %post

```
Bootstrap: debootstrap
MirrorURL: http://archive.ubuntu.com/ubuntu/
OSVersion: bionic
Include: software-properties-common
```

OR

```
Bootstrap: localimage
From: /path/to/image
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
- **%files**
- %environment
- %post

```
%files
        /file1
        /file2 /opt
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
- %files
- **%environment**
- %post

```
%environment
        export PATH=/opt/anaconda3/bin:${PATH}
        export INSTALL_DIR=/opt/installer
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
- %files
- %environment
- **%post**

```
%post
        export ANACONDA_FILE=Anaconda3-5.2.0-Linux-x86_64.sh

        apt-add-repository universe
        apt-get update –y
        apt-get install –y wget git libfftw3

        wget https://repo.continuum.io/archive/${ANACONDA_FILE}
        bash ${ANACONDA_FILE} -b -p /opt/anaconda3
        export PATH=/opt/anaconda3/bin:${PATH}

        pip install astropy scipy scikit-learn

        apt-get clean
        rm ${ANACONDA_FILE}
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
- %files
- %environment
- %post
- **%runscript, %app, %label, %help,**

```
%runscript
    python "$@"
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

- Header
- %files
- %environment
- %post
- %runscript

```
Bootstrap: debootstrap
MirrorURL: http://archive.ubuntu.com/ubuntu/
OSVersion: bionic
Include: software-properties-common

%environment
          export PATH=/opt/anaconda3/bin:${PATH}

%post
          export ANACONDA_FILE=Anaconda3-5.2.0-Linux-x86_64.sh

          apt-add-repository universe
          apt-get update –y
          apt-get install –y wget git libfftw3

          wget https://repo.continuum.io/archive/${ANACONDA_FILE}
          bash ${ANACONDA_FILE} -b -p /opt/anaconda3
          export PATH=/opt/anaconda3/bin:${PATH}

          pip install astropy scipy scikit-learn

          apt-get clean
          rm ${ANACONDA_FILE}

%runscript
          python "$@"
```

Reference: https://sylabs.io/guides/3.5/user-guide/

Singularity – Definition file

```
singularity build <container_name> /path/to/definition_file
```

```
singularity build ML-astro.simg ML-astro.def
```

Containers:

- .img, .simg, .sif
- No difference, just naming conventions that have changed over time

- Contact support@ilifu.ac.za for assistance

Reference: https://sylabs.io/guides/3.5/user-guide/

Custom Jupyter kernel

Available kernels:

```
jupyter-kernelspec list
```

Location of custom user kernel:

```
/users/<username>/.local/share/jupyter/kernel/<kernel_name>/kernel.json
```

Example:

```
/users/jeremy/.local/share/jupyter/kernel/ML/kernel.json
```

Custom Jupyter kernel

kernel.json:

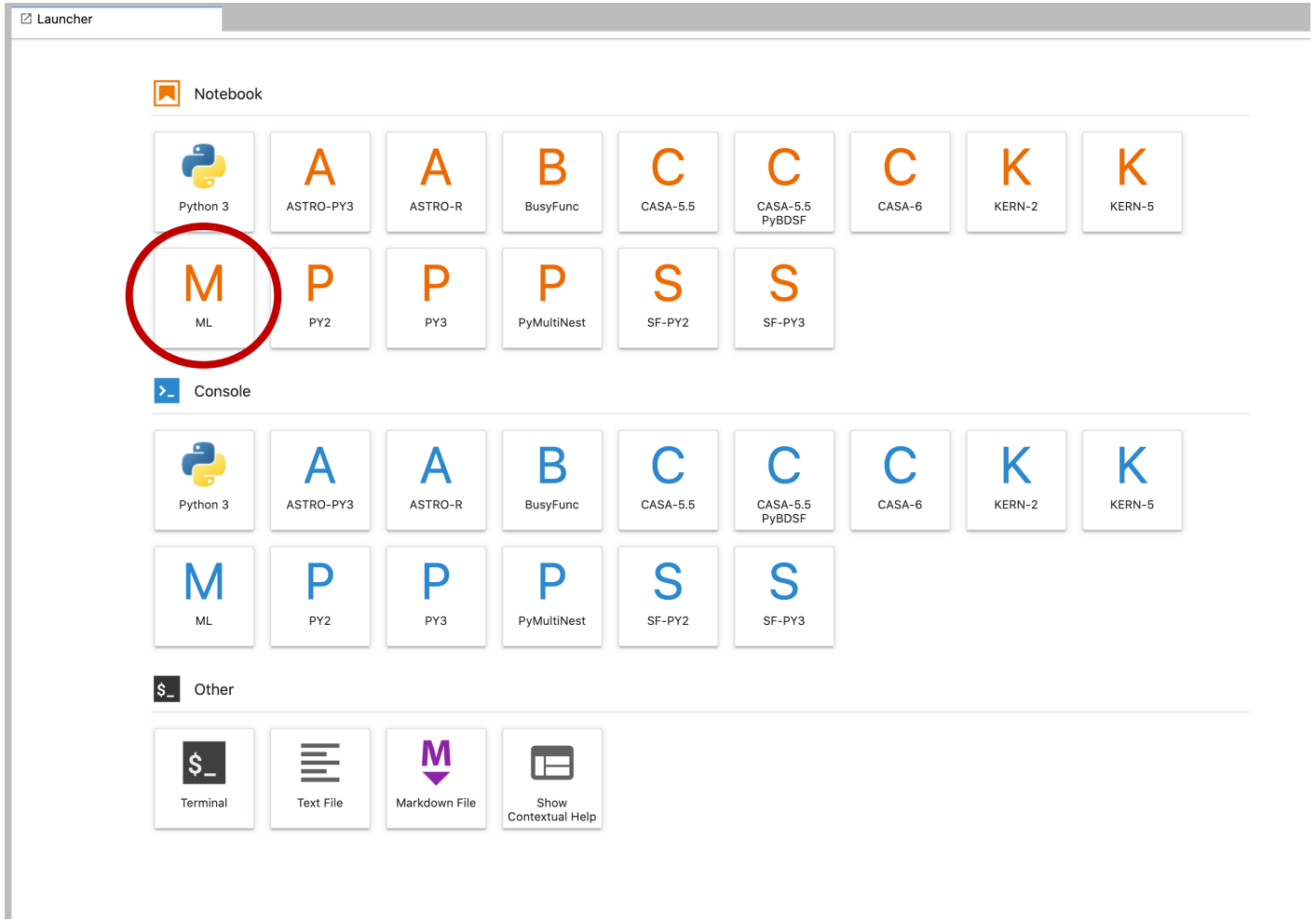
```
{
 "display_name": ”<kernel_name>",
 "language": "python",
 "argv": [
  "singularity", "exec", "/path/to/container", ”/path/to/python_exec",
  "-m",
  "IPython.kernel",
  "-f",
  "{connection_file}"
 ]
}
```

Custom Jupyter kernel

Example kernel.json:

```
{
 "display_name": ”ML",
 "language": "python",
 "argv": [
  "singularity", "exec", "/idia/users/jeremy/containers/ML.simg", ”/anaconda3/bin/python",
  "-m",
  "IPython.kernel",
  "-f",
  "{connection_file}"
 ]
}
```

Custom Jupyter kernel



Python Virtual Environment

- virtualenv
- https://virtualenv.pypa.io/en/latest/
- Isolated Python environment
- Less risk of conflicts occurring with pip install --user
- Similar to venv (python -m venv)
- Can customize which os python is used: python2.7, python3+
- Limited by os libraries

Reference: https://virtualenv.pypa.io/en/latest/

Python Virtual Environment

```
virtualenv --help
virtualenv /path/to/virtual_environment
```

`--python`	The Python interpreter to use
`--system-site-packages`	Give the virtual environment access to the global site-packages

Reference: https://virtualenv.pypa.io/en/latest/

Python Virtual Environment

```
virtualenv /path/to/virtual_environment
```

Example:

```
virtualenv /idia/users/jeremy/venv/ML
source /idia/users/jeremy/venv/ML/bin/activate
which python
    ➢ /idia/users/Jeremy/venv/ML/bin/python
pip install torch scikit-learn
deactivate (to exit)
```

Reference: https://virtualenv.pypa.io/en/latest/

Python Virtual Environment

Python virtualenv as a Jupyter kernel

Once the virtual environment is active:

```
ipython kernel install --name "<kernel_name>" --user
```

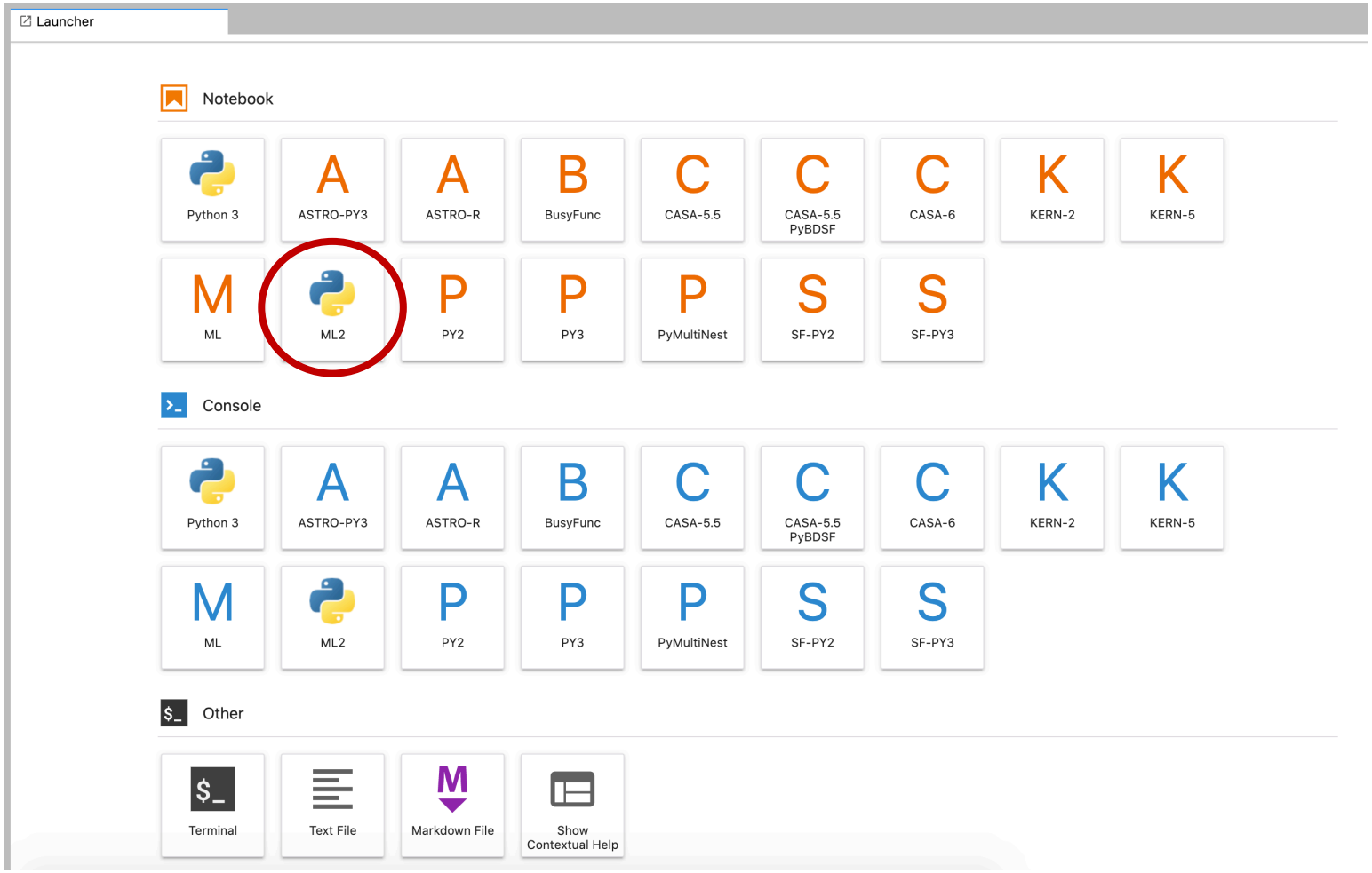
Example:

```
source /idia/users/jeremy/venv/ML/bin/activate
pip install jupyter
ipython kernel install --name "ML2" --user
```

Creates the kernel.json file at
`/users/jeremy/.local/share/jupyter/kernels/ml2/kernel.json`

Reference: https://virtualenv.pypa.io/en/latest/

Python Virtual Environment



Reference: https://virtualenv.pypa.io/en/latest/