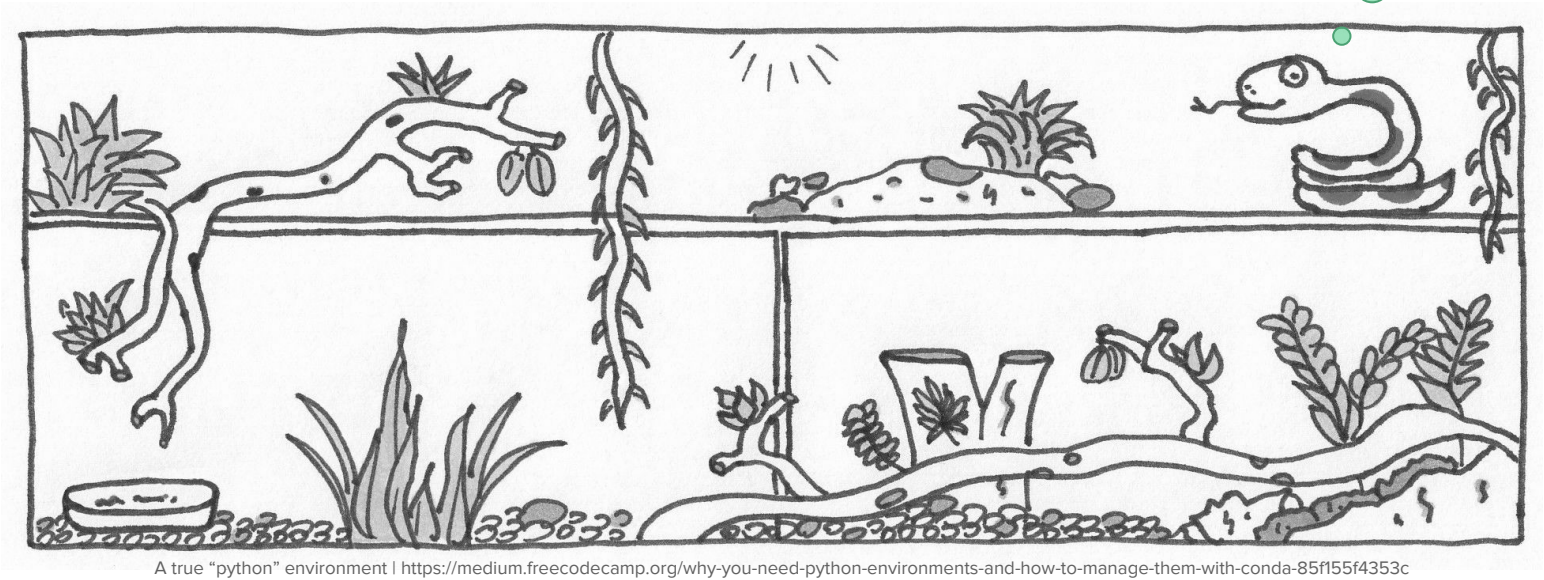
ACCRE Environments

SDT | 22 May 2018

What is a virtual environment?

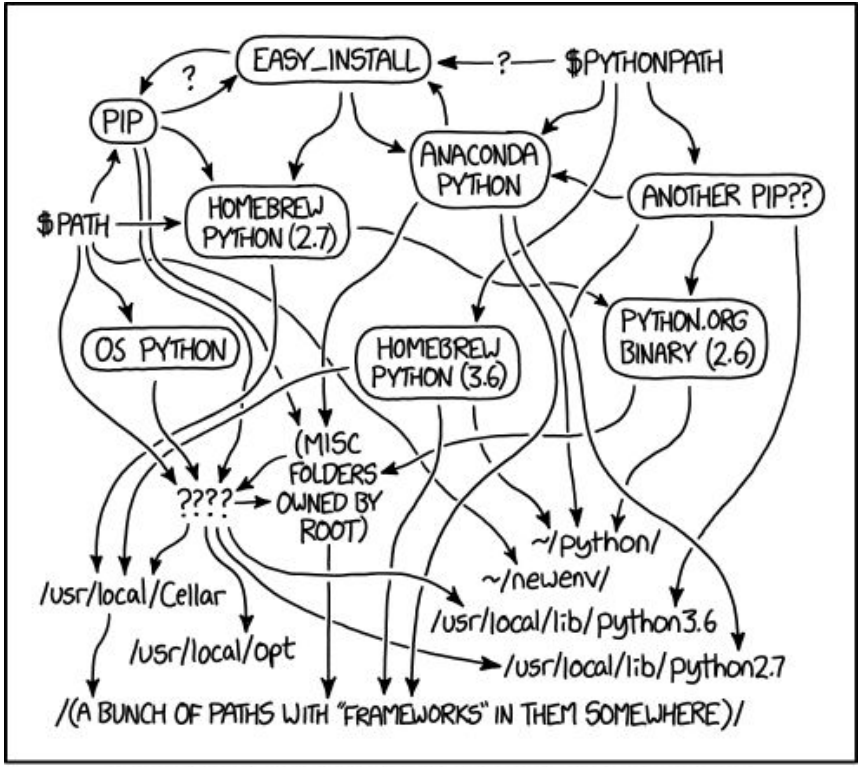
- Isolated space with its own files, paths, libraries, dependencies
- The **`conda environment`** is a directory holding a set of installed packages



A true “python” environment | https://medium.freecodecamp.org/why-you-need-python-environments-and-how-to-manage-them-with-conda-85f155f4353c

Yes. You really need one.

- Manage complex dependencies
- Install new packages with ease
- Record all installed packages
- Run specific software/Python versions
- Simplify collaboration
- Avoid installation permission issues
- Maintain code reproducibility
- Isolate packages from each other & the global environment



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

https://m.xkcd.com/1987/

Anaconda to the rescue

- Organized Python packages for data science and machine learning
- The **conda** package is a package, dependency and environment manager for any language

Lmod: the basics

Load Anaconda before creating/activating a conda environment:

```
module load Anaconda3
```

.....

KINDA LIKE AN ACCRE VIRTUAL ENV...

Save a specific module configuration with:

```
module save [ name ]
```

Load a saved module configuration with:

```
module restore [ name ]
```

ALSO:
MODULE SAVELIST
MODULE DESCRIBE

How do I create an environment?

- Default create: **`conda create --name gallifrey`**
  - Environment will contain Python 2.7 if using Anaconda2
- Specify Python version: **`conda create --name gallifrey python=3.5`**

```
[colbrall@chgr1 ~]$ conda create --name gallifrey
Fetching package metadata ...........
Solving package specifications:
Package plan for installation in environment /home/colbrall/.conda/envs/gallifrey:

Proceed ([y]/n)? y

#
# To activate this environment, use:
# > source activate gallifrey
#
# To deactivate this environment, use:
# > source deactivate gallifrey
#
```

Which environments do I have?

- **`conda env list`**

```
[colbrall@chgr1 ~]$ conda env list
# conda environments:
#
R3.4                     /home/colbrall/.conda/envs/R3.4
gallifrey                /home/colbrall/.conda/envs/gallifrey
julia0.6.2               /home/colbrall/.conda/envs/julia0.6.2
root                  *  /opt/easybuild/software/Core/Anaconda2/4.4.0
```

- Alternatively: **`ls .conda/envs/`**

```
[colbrall@chgr1 ~]$ ls .conda/envs/
gallifrey  julia0.6.2  R3.4
[colbrall@chgr1 ~]$
```

Activating and Deactivating environments

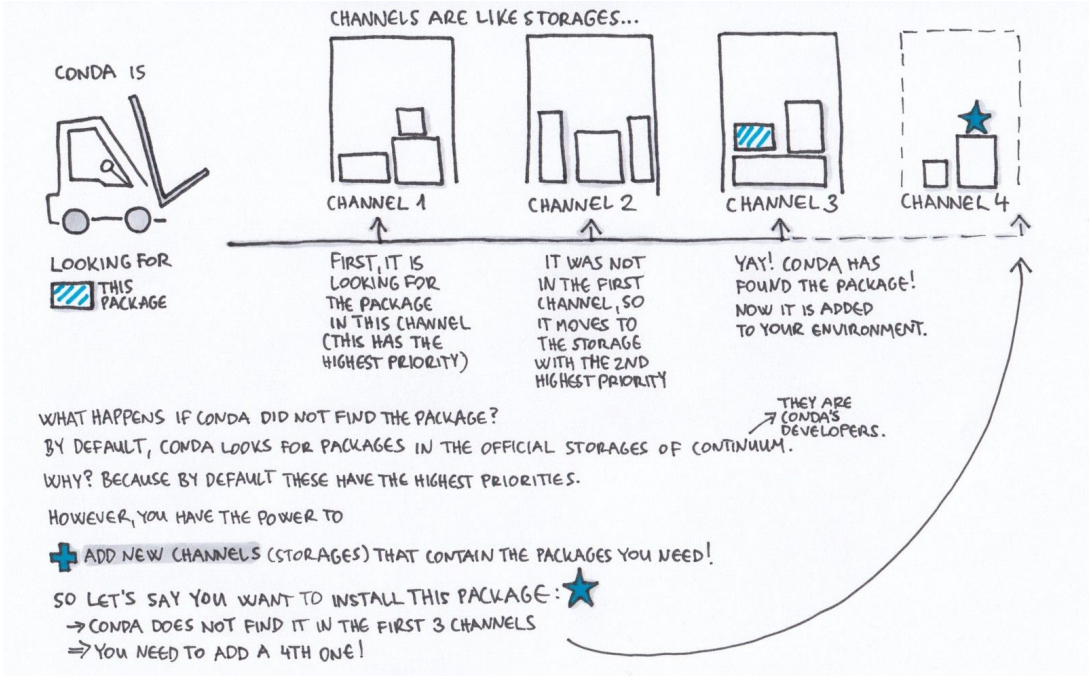
- **`source activate gallifrey`**

```
[colbrall@chgr1 ~]$ source activate gallifrey
(gallifrey) [colbrall@chgr1 ~]$ ls
00TEMPLATE.slurm  classes  data  intel  projects
(gallifrey) [colbrall@chgr1 ~]$
```

- **`source deactivate gallifrey`**

```
(gallifrey) [colbrall@chgr1 ~]$ source deactivate
[colbrall@chgr1 ~]$ ls
00TEMPLATE.slurm  classes  data  intel  projects
[colbrall@chgr1 ~]$
```

Installing/managing packages: channels



From: https://medium.freecodecamp.org/why-you-need-python-environments-and-how-to-manage-them-with-conda-85f155f4353c

After conda 4.1.0, switch to channel priority > version.

We may change this behaviour with

```
conda config --set channel_priority false
```

Managing packages: channels

- `conda config --get channels`
- `Conda config --append channels bioconda`
- `Conda config --prepend/--add channels conda-forge`
- `Conda config --remove channels conda-forge`

```
(gallifrey) chenl11 ~ $
conda config --get channels
--add channels 'bioconda'   # lowest priority
--add channels 'defaults'   # highest priority
(gallifrey) chenl11 ~ $
conda config --append channels bioconda
Warning: 'bioconda' already in 'channels' list, moving to the bottom
(gallifrey) chenl11 ~ $
conda config --prepend channels conda-forge
(gallifrey) chenl11 ~ $
conda config --get channels
--add channels 'bioconda'   # lowest priority
--add channels 'defaults'
--add channels 'conda-forge'   # highest priority
(gallifrey) chenl11 ~ $
conda config --remove channels conda-forge
(gallifrey) chenl11 ~ $
conda config --get channels
--add channels 'bioconda'   # lowest priority
--add channels 'defaults'   # highest priority
```

Managing packages: search

```
conda search [-c r] -f r-irkernel
```

```
(gallifrey) chenl11 ~ $
conda search -c r -f r-irkernel
Fetching package metadata .............
r-irkernel                    0.2                r3.1.3_0  defaults
                              0.2               r3.1.3_0a  defaults
                              0.2                r3.1.3_1  defaults
                              0.2               r3.1.3_1a  defaults
                              0.3                r3.1.3_0  defaults
                              0.3               r3.1.3_0a  defaults
                              0.4                r3.1.3_0  defaults
                              0.4                r3.2.1_0  defaults
                              0.4               r3.1.3_0a  defaults
                              0.4               r3.2.1_0a  defaults
                              0.4                r3.2.1_1  defaults
                              0.4                r3.2.2_1  defaults
                              0.4               r3.2.1_1a  defaults
                              0.4               r3.2.2_1a  defaults
                              0.5                r3.2.2_1  defaults
                              0.5               r3.2.2_1a  defaults
                              0.5                r3.2.2_2  defaults
                              0.6.1              r3.3.1_0  defaults
                              0.7                r3.3.1_0  defaults
                              0.7                r3.3.1_2  defaults
                              0.7.1              r3.3.2_0  defaults
                           *  0.7.1              r3.4.1_0  defaults
                              0.8.9          r342hfe3cb8f_0  defaults
                              0.8.11             mro343_0  defaults
                              0.8.11               r343_0  defaults
                              0.2                r3.1.3_0  r
                              0.2               r3.1.3_0a  r
                              0.2                r3.1.3_1  r
                              0.2               r3.1.3_1a  r
                              0.3                r3.1.3_0  r
                              0.3               r3.1.3_0a  r
                              0.4                r3.1.3_0  r
                              0.4                r3.2.1_0  r
                              0.4               r3.1.3_0a  r
                              0.4               r3.2.1_0a  r
                              0.4                r3.2.1_1  r
                              0.4                r3.2.2_1  r
                              0.4               r3.2.1_1a  r
                              0.4               r3.2.2_1a  r
                              0.5                r3.2.2_1  r
                              0.5               r3.2.2_1a  r
                              0.5                r3.2.2_2  r
                              0.6.1              r3.3.1_0  r
                              0.7                r3.3.1_0  r
                              0.7                r3.3.1_2  r
                              0.7.1              r3.3.2_0  r
                              0.7.1              r3.4.1_0  r
                              0.8.9          r342hfe3cb8f_0  r
                              0.8.11             mro343_0  r
                              0.8.11               r343_0  r
```

Managing packages: install and remove

conda install [-n gallifrey] [-c r] r-irkernel

```
(gallifrey) chenl11 ~ $
conda install -n gallifrey -c r -f r-irkernel
Fetching package metadata .............
Solving package specifications: .

Package plan for installation in environment /scratch/chenl11/conda_envs/gallifrey:

The following NEW packages will be INSTALLED:

    r-irkernel: 0.7.1-r3.4.1_0 r

Proceed ([y]/n)? y
```

conda remove r-irkernel

```
(gallifrey) chenl11 ~ $
conda list | grep r-irkernel
r-irkernel                0.7.1                r3.4.1_0    r
(gallifrey) chenl11 ~ $
conda remove r-irkernel
Fetching package metadata ...........
Solving package specifications: .

Package plan for package removal in environment /scratch/chenl11/conda_envs/gallifrey:

The following packages will be REMOVED:

    r-irkernel: 0.7.1-r3.4.1_0 r

Proceed ([y]/n)? y

(gallifrey) chenl11 ~ $
conda list | grep r-irkernel
(gallifrey) chenl11 ~ $
```

Share an environment

```
conda env export > gallifrey_env.yml
```

```
(gallifrey) chenl11 ~ $
conda env export > gallifrey_env.yml
(gallifrey) chenl11 ~ $
more gallifrey_env.yml
name: gallifrey
channels:
- defaults
- bioconda
dependencies:
- _r-mutex=1.0.0=anacondar_1
- backports=1.0=py27_0
- backports_abc=0.5=py27_0
- bleach=1.5.0=py27_0
- bzip2=1.0.6=3
```

```
conda env create -f environment.yml
```

Delete an environment

```
conda env remove --name gallifrey
```

And don’t forget to clean

Conda doesn’t remove packages when you remove an environment, just the structure to access them.

`conda clean --all` removes any packages, caches, etc. that aren’t used by any of your environments

Useful for staying under ACCRE file limits!

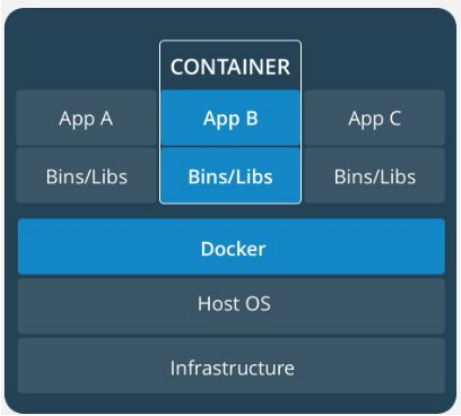
Conda is so powerful, why others?

Conda is good at managing packages for programming languages, but is not ideal to manage other dependencies.

Imagine we have to deploy a work pipeline with specific version of software, for instance, BEDTools/2.26.0. How can we ensure a reproducible, portable and immutable environment?

One solution is using a container.

What is a container?



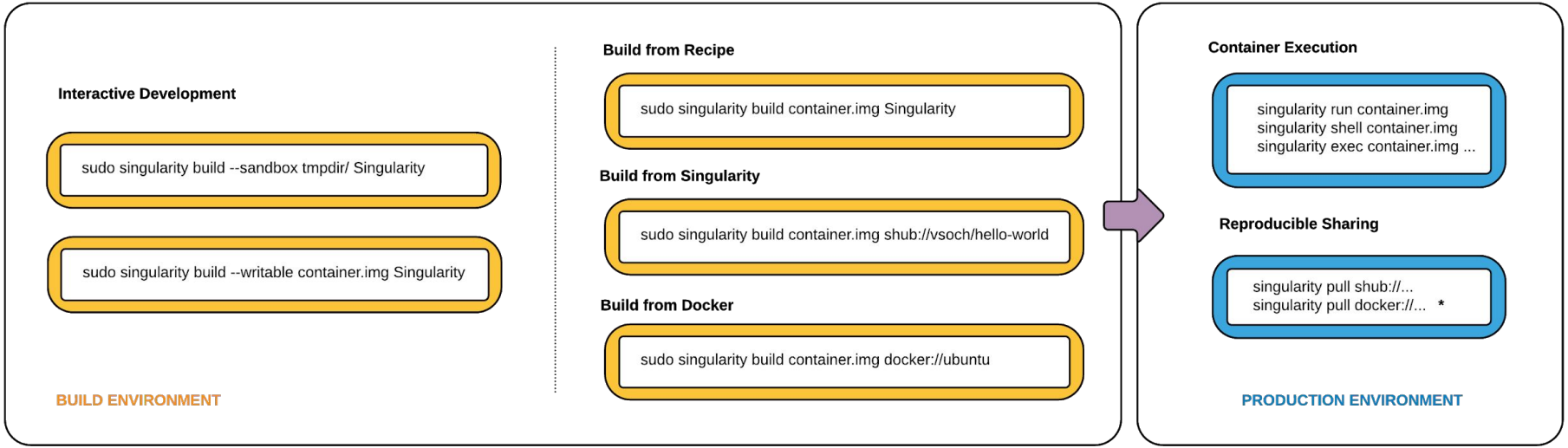
A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.

A HPC container Singularity

Compare to docker, Singularity is designed for HPC computing. Because

- Security reasons. Docker gives root privileges. It’s hard to limit access.
- Scheduling. It’s hard to schedule resources to docker containers. On the other hand, slurm is built in to singularity.
- Singularity has better integration to *MPI*, GPUs…

The singularity workflow



* Docker construction from layers not guaranteed to replicate between pulls

The fastest way to use a singularity container

```
singularity shell shub://vsoch/hello-world
```

```
chenl11 /dors/capra_lab/users/chenl/projects $
singularity shell shub://vsoch/hello-world
Progress |==================================| 100.0%
WARNING: Non existent bind point (directory) in container: '/scratch'
WARNING: Non existent bind point (directory) in container: '/data'
WARNING: Non existent bind point (directory) in container: '/dors'
WARNING: Non existent bind point (directory) in container: '/gpfs22'
WARNING: Non existent bind point (directory) in container: '/gpfs23'
WARNING: Non existent 'bind path' source: '/legodata'
Singularity: Invoking an interactive shell within container...

Singularity vsoch-hello-world-master-latest.simg:~> whoami
chenl11
Singularity vsoch-hello-world-master-latest.simg:~>
```

A more advanced example: a singularity recipe

```
Bootstrap: docker
From: ubuntu

%help
Help me. I'm in the container.

%setup
  touch ${SINGULARITY_ROOTFS}/tacos.txt
  echo "I love avocados!" > avocados.txt

%files
  avocados.txt

%labels
  Maintainer Ling
  Version v1.0

%environment
  CAT=genius
  export CAT

%post
  mkdir /dors /scratch /data /gpfs22 /gpfs23

  apt-get update
  apt-get -y --force-yes install vim \
                                 python-numpy \
                                 python-pip && \
                                 apt-get clean


%runscript
  echo "Hello world from the container!"
  echo "Argument received: $*"
  exec echo "$@"
```

Header: Define where to get the OS

Setup: Execute outside of container, after the base OS is installed.

Files: cp files into containers

Post: Execute inside of container at build time. Meat of setup will live here. Install the packages you want here.

Runscript: Execute when the container is run, not at build time. Can take arguments.

Test it out!

```
vagrant@vagrant-ubuntu-trusty-64:~$ sudo singularity build test.img test.def
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$ singularity help test.img


Help me. I'm in the container.

vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$ singularity exec test.img cat avocados.txt
WARNING: Non existent bind point (file) in container: '/etc/localtime'
I love avocados!
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$ singularity exec test.img env | grep CAT
WARNING: Non existent bind point (file) in container: '/etc/localtime'
CAT=genius
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$
vagrant@vagrant-ubuntu-trusty-64:~$ singularity run test.img you shake it all about
WARNING: Non existent bind point (file) in container: '/etc/localtime'
Hello world from the container!
Argument received: you shake it all about
you shake it all about
```

Useful Links

Anaconda site: https://www.anaconda.com/download/

Conda cheatsheet: https://conda.io/docs/_downloads/conda-cheatsheet.pdf

Conda-forge package list: https://conda-forge.org/feedstocks/ (includes R, Julia packages)

Official Guide: https://conda.io/docs/user-guide/getting-started.html