conda Environments Basics

ANACONDA®

Environments

If you are going to be sharing your work with others, you need to be aware of managing dependencies.

Dependencies are the libraries, including the *versions* of libraries that you have installed.

Environments

Why do you have to worry about dependencies?

- Libraries release new versions, which add, deprecate, or remove functions.

- Particular versions of libraries may not work well together.

Version 1.0

- What's new in 1.0.5 (June 17, 2020)
  - Fixed regressions
  - Bug fixes
  - Contributors
- What's new in 1.0.4 (May 28, 2020)
  - Fixed regressions
  - Bug fixes
  - Contributors
- What's new in 1.0.3 (March 17, 2020)
  - Fixed regressions
  - Bug fixes
  - Contributors
- What's new in 1.0.2 (March 12, 2020)
  - Fixed regressions
  - Indexing with Nullable Boolean Arrays
  - Bug fixes
  - Contributors
- What's new in 1.0.1 (February 5, 2020)
  - Fixed regressions
  - Deprecations
  - Bug fixes
  - Contributors
- What's new in 1.0.0 (January 29, 2020)
  - New Deprecation Policy
  - Enhancements
  - Experimental new features
  - Other enhancements
  - Backwards incompatible API changes
  - Deprecations

Release Notes for the pandas Library

Package Managers

The two most common package managers for installing third party packages you'll use are *pip* and *conda*.

pip: The standard package manager for Python. Installs packages from the Python Package Index, aka PyPI, aka the Cheese Shop, the official third-party repository for Python.

conda: A package manager designed for use by data analysts/data scientists. Installs packages from Anaconda's repository. When installing, will analyze the current environment and ensure that there are no conflicts between package versions.

conda

It is recommended that you install packages using **conda** whenever possible and use pip only when a package is not available from conda.

See https://www.anaconda.com/blog/using-pip-in-a-conda-environment for more information.

conda

conda is not just a package manager, but is also an *environment* manager, meaning that you can create separate environments containing files, packages, their dependencies, and their own versions of the Python interpreter.
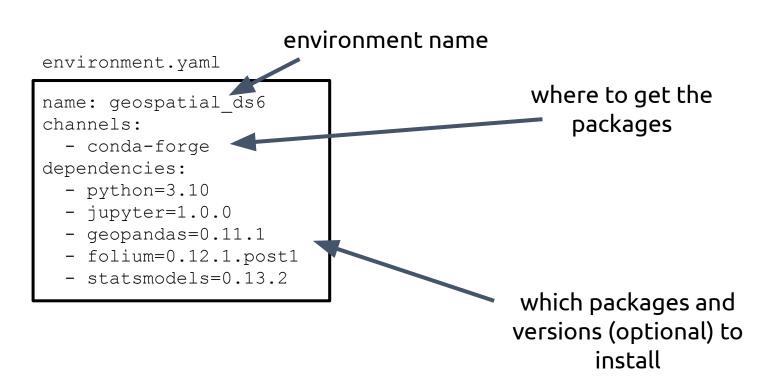
This serves two purposes:

1. Isolates your projects

2. Makes it easier to share your work and allows for reproducibility.

conda environments

Two ways to create a conda environment:

1. Start from scratch:

   ```
   $ conda create -n <environment name> python=<version>
   ```

```
(base) michael@michael-HP-Pavilion-Laptop-15-cs3xxx ~ $ conda create -n testenv
python=3.8
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/michael/anaconda3/envs/testenv

  added / updated specs:
    - python=3.8


The following NEW packages will be INSTALLED:

  _libgcc_mutex       pkgs/main/linux-64::_libgcc_mutex-0.1-main
  ca-certificates     pkgs/main/linux-64::ca-certificates-2020.6.24-0
  certifi             pkgs/main/linux-64::certifi-2020.6.20-py38_0
  ld_impl_linux-64    pkgs/main/linux-64::ld_impl_linux-64-2.33.1-h53a641e_7
```

conda Environments

2. Create from a YAML file which lists which packages and versions should be installed.

environment name

environment.yaml

```
name: geospatial_ds6
channels:
  - conda-forge
dependencies:
  - python=3.10
  - jupyter=1.0.0
  - geopandas=0.11.1
  - folium=0.12.1.post1
  - statsmodels=0.13.2
```

where to get the packages

which packages and versions (optional) to install

conda Environments

To create an environment from a YAML file, run this from the folder containing the environment file:

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

conda Environments

You can see a list of all of your conda environments along with the one currently active by typing

```
$ conda env list
```


This will display a * next to the active environment.

conda Environments

To switch environments, type

```
$ conda activate <environment name>
```

```
(base) michael@michael-HP-Pavilion-Laptop-15-cs3xxx ~ $ conda activate geospatial
(geospatial) michael@michael-HP-Pavilion-Laptop-15-cs3xxx ~ $ conda env list
# conda environments:
#
base                     /home/michael/anaconda3
geospatial            *  /home/michael/anaconda3/envs/geospatial
```

If you need to return to the base environment, type

```
$ conda deactivate
```

conda Environments

Then (as long as your current environment includes jupyter), you can launch jupyter in the current environment by typing

```
$ jupyter notebook
```

Installing packages

Then to install in the **current environment**

```
$ conda install <package name>
```

For a specific version:

```
$ conda install <package name>=<version number>
```

conda

Download a conda cheat sheet here:
https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c496712
67e40c689e0bc00ca/conda-cheatsheet.pdf

For more information, see also the DataCamp course *Conda Essentials:*
https://learn.datacamp.com/courses/conda-essentials