Miniconda 101

Package, dependency and environment management for any language---Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN

https://docs.conda.io/projects/conda/en/latest/

Configuration for NVU-Lyndon Atmospheric Sciences (ASAC 315 & WX Center)

Conda is an open source package management system and environment management system that runs on Windows, macOS and Linux. Conda quickly installs, runs and updates packages and their dependencies. Conda easily creates, saves, loads and switches between environments on your local computer. It was created for Python programs, but it can package and distribute software for any language.

Common Python packages applicable to Atmospheric Sciences that can be installed with Miniconda include (but are not limited to): numpy, CDO, WRF-Python, wrfskewt, metpy, python-awips, jupyter, and QGIS.

Conda as a package manager helps you find and install packages. If you need a package that requires a different version of Python, you do not need to switch to a different environment manager, because conda is also an environment manager. With just a few commands, you can set up a totally separate environment to run that different version of Python, while continuing to run your usual version of Python in your normal environment.

Miniconda is “mini” because by default, it does not include as many Python packages as Anaconda does. A minimum number of default packages uses less space on hard drives.

Starting Miniconda on Windows

1. Click on the Start menu, and search for **Anaconda**. Click the **Anaconda Prompt** to open the anaconda prompt.

Starting Miniconda on CentOS Linux

1. Miniconda is not installed by default on Linux. If you do not have Miniconda installed, please see Jason Kaiser to have him quickly install Miniconda for you (to your Linux home directory).

   While it is possible to install Miniconda yourself, it will likely take 15+ minutes to do so. If you are not sure if you have Miniconda installed, see step 1, page 2 of this documentation to test your Miniconda installation.

Test Miniconda, Install Conda-Forge Channel, and Update Conda to the Latest Version

1. To test your Miniconda installation, type the following into the Anaconda Prompt (on Windows) or the Terminal window (on Linux):

   ```
   conda info
   ```

   If you see “conda not found” when running this command, Miniconda is not installed.

   Special note about Miniconda on Windows

   When typing **conda info** the ”active env location” should list **C:\Miniconda3**.This location is chosen by default on Windows in order to allow multiple users on the same computer to have permissions to install new packages.
   See page 3 of this documentation for the importance regarding how to create environments and why you need to be careful managing environments at this shared **C:\Miniconda3** location.

2. Type the following into the Anaconda Prompt (on Windows) or the Terminal window (on Linux) to install the conda-forge channel. You should run this command even if you have done so previously, as it will ensure that conda-forge is set as the highest priority channel.

   ```
   conda config --add channels conda-forge
   ```

   About Channels

   Installing new channels allows you to then install additional Python packages that would not otherwise be available. The conda-forge channel is a community-driven collection of packages that are continually tested to ensure compatibility.
   Please refer to: https://docs.anaconda.com/anaconda/navigator/tutorials/manage-channels/ for additional documentation on managing channels.

3. Update conda to the latest version. Type the following into the Anaconda Prompt (on Windows) or the Terminal window (on Linux), and press **y** to confirm the packages to be updated (even if you just installed Miniconda, there may be updates available. Conda developers recommend that you always keep conda updated to the latest version.)

   ```
   conda update conda
   ```

   Note about entering commands

   Windows is not case-sensitive, while Linux is always case-sensitive. When in doubt, use lowercase conda commands on both Windows and Linux.

Managing Conda Environments, Installing Packages, and Best Practices

About Environments

With conda, you can create, export, list, remove, and update environments that have different versions of Python and/or packages installed in them. Switching or moving between environments is called activating the environment.

Additional conda environments resources:

- https://docs.anaconda.com/anaconda/navigator/tutorials/manage-channels/ (more info not included in this documentation)
- **Unidata's MetPy Mondays video on conda environments:** https://youtu.be/15DNH25UCi0

1. By default, Miniconda loads the "base" environment. **It is a best practice to create a new conda environment and install any new Python packages to this new environment.** A new, unique, conda environment allows you to install specific versions of Python packages (and their dependencies – additional Python packages that conda automatically determines are necessary for the Python packages [and versions] you've specified) to the environment name that you specify. Installing Python packages to unique environments eliminates the possibility for package version conflicts (for example, if you install SHARPpy with numpy 1.9.3, but also install metpy to the same environment, metpy may require a newer version of numpy, creating a package version conflict).

2. To list conda environments that exist, type the following into the Anaconda Prompt (on Windows) or the Terminal window (on Linux):

```
conda info --envs
```

   Specifically on Windows, running this command may show you environments that other users have already created (because the C:\Miniconda3 location is a shared location for multiple environments). The environment with an asterisk (*) next to it is the currently active environment.

3. To activate a **preexisting** environment, type the following in the Anaconda Prompt (on Windows) or the Terminal window (on Linux):

```
conda activate myenv
```

   Replace *myenv* with the desired environment name. By default, the active environment---the one you are currently using---is shown in parentheses () or brackets [] at the beginning of your command prompt. **If you're on Windows, please follow this convention when naming environments (due to the shared nature of environments among multiple users):**

Environment Name	Packages Installed
sharppyenv	pyside=1.2.4 numpy=1.9.3 (and dependencies)
cdoenv	cdo (and dependencies)
wrfpythonenv	wrf-python jupyter matplotlib cartopy netCDF4 (and dependencies)
qgisenv	qgis (and dependencies)
wrfskewtenv	wrfskewt (and dependencies)

If you're on Linux, you can name the environment(s) any name you'd like (since the environments will be created under your own Linux home directory), as long as it is fairly descriptive so that you can recognize the environment name in the future.

4. If you don't already see the environment that you want in the preexisting list of environments, to create a **new** environment, and install the desired package(s) to this environment, type the following into the Anaconda Prompt (on Windows) or the Terminal window (on Linux):

Main package to install	**Environment creation and package install command (type one of these lines into the prompt)**
SHARPpy (packages for)	`conda create -n sharppyenv pyside=1.2.4 numpy=1.9.3`
cdo	`conda create -n cdoenv cdo`
wrf-python	`conda create -n wrfpythonenv jupyter matplotlib cartopy netCDF4 wrf-python`
qgis	`conda create -n qgisenv qgis`
wrfskew	`conda create -n wrfskewtenv wrfsket`
python-awips	***

*** for Python-AWIPS, follow the directions for a conda environment install of Python-AWIPS at https://python-awips.readthedocs.io/en/latest/index.html

5. Activate your new environment by typing:

```
conda activate myenv
```

Replace *myenv* with the desired environment name. By default, the active environment---the one you are currently using---is shown in parentheses () or brackets [] at the beginning of your command prompt.

6. That's it! You're ready to start working with Python packages installed to your currently-active environment. From here, follow the directions for:
   - Additional documentation listed at the bottom of this page: http://atmos.northernvermont.edu/student-resources/
     - For SHARPpy on Windows documentation, start at step 5
     - For WRF-Python on Windows documentation, start at step 5
     - For CDO on Linux, start at step 11

For Future Reference

Anaconda (and Miniconda) have a Cheat Sheet for conda commands (some of which are not listed in this documentation).

Refer to: http://know.continuum.io/rs/387-XNW-688/images/conda-cheatsheet.pdf

If you experience any trouble, error messages, etc. following this documentation, please contact Jason Kaiser.