Understanding a Canopy system-wide installation (--common-install)

Updated: 5/17/2016

The Canopy Command Line interface (CLI) includes powerful features for system administrators to create multi-user and automated Canopy installations. We are working on expanding and clarifying the CLI documentation.

Meanwhile, this article provides some additional discussion of one of the most popular CLI use cases -- a system-wide installation (common install).

Overview

The key advantages of a common install, in contrast to a default shared installation with multiple users, are:

- Updating library packages is usually done by the central administrator on behalf of all users (rather than by each individual user). This is particularly useful when the users are not skilled Python developers or when you want all users to use the same package versions.
- The Python environments which Canopy creates for each user are very lightweight (rather than hundreds of MB per user). This is particularly useful when there are constraints on the size of user home directories (roaming profiles or limited disk storage).

A common install results in a 3-level set of virtual environments:

Level 0: A Canopy Core base environment, installed by an administrator (from the MSI on Windows, .dmg on Mac, or shell script on Linux), on either a local machine or a central server.

Level 1: A common System (GUI application) virtual environment and User Python virtual environment, inheriting from Level 0, created by the administrator using the Canopy Command Line Interface (CLI), on either a local machine or a central server.

Level 2: A System and User environment for each user, inheriting from Level 1, created (by default) in each user's home directory the first time they start Canopy; these are very lightweight.
Procedure

Please begin by reading

- "Where are all of the Python packages in my User Python Environment?" for an overview of Canopy's use of virtual environments,
- "Scenario: Creating a system-wide Canopy install" in the CLI documentation for a description of this type of installation.

Windows

1) Install the Canopy MSI for all users. This installs Canopy Core (Level 0). You can refer, if needed, to this Knowledge Base article: "Windows - on some systems, admin users cannot immediately install "for all users"."

If you are installing the 1 GB Canopy Full installer, this MSI installation step may take as long as 10 minutes depending on your hardware.

In the dialog window at the end of the installation, be sure to uncheck the box for "Launch application".

If, as described in the above article, you need to install the MSI in a command prompt window with escalated administrative privileges, and you intend also to place the common install into Program Files as recommended (step 3 below), then leave the "Administrator: Command Prompt" window open, because you will need it for that step.

2) Do not run the Canopy application yet, as this would create (though reversibly) a configuration that is incompatible with a common install. (If you did run the application prematurely, delete the Enthought subdirectories from your own (hidden) %APPDATA% and %LOCALAPPDATA% directory (usually "C:\Users\<user>\AppData\Roaming\Enthought", and "...
AppData\Local\Enthought", and proceed.)

3) From within the Canopy MSI installation directory (Canopy Core), run the system-wide installation (common install):

`Canopy_cli.exe --common-install --install-dir=<target directory>`

This step is only run once, by one admin user whom we will call "the admin" (even though some other users may also have admin credentials). After a dialog to confirm your choice of install dir, this command creates common System (Canopy GUI app) and User Python environments in a Canopy or Canopy32 subdirectory of the specified install-dir. These environments will be
used and updated directly by the admin; all other users will subsequently inherit from these environments but will not be able to update them.

We recommend, in most cases, if you installed Canopy Core into "C:\Program Files\Enthought\Canopy\App" (the default), that you locate the common System and User directories as siblings to Canopy Core, by using the following common install command:

cd "C:\Program Files\Enthought"
Canopy\App\Canopy_cli.exe --common-install --install-dir=. 

As mentioned in step 1 above, you may need to run this in a command prompt window with elevated administrative privileges.

If you installed the Full installer in step 1, this step may take as long as 20 minutes, depending on your hardware.

4) After the common System and User environments are created, the final dialog will ask "Do you want to make Canopy your default Python environment?" Explicitly select "No". Then after you click "Start Using Canopy", the Canopy application normally opens; in this common-install scenario, Canopy sometimes does not open. In this case, open it from the (public) Canopy desktop icon.

5) The first time that each user (other than the admin) starts Canopy from the Windows Start Menu, his Python environments will automatically be created, inheriting from the admin’s common install.

MacOS & Linux (Canopy 64-bit)

Note: To avoid any conflict with system Python, this installation and one-time setup is done almost entirely not as root, but rather by a user whom we will call the "admin" for the purpose of installing and maintaining the Canopy system-wide installation. Depending on your target directories, you may need to run as root briefly to create the target directories and set their group and owners so that the "admin" will have full write access to them, and all other Canopy users will have read and execute access, but do not stay as root for the installation and setup steps which follow.

For this example, you first take root to create /usr/local/CanopyCore/ and /usr/local/Canopy_64bit/ with full write access for the "admin", and read/execute access for the regular Canopy users. Then the following steps are run by the "admin", not as root (e.g. not using sudo with default parameters):
1) Install Canopy Core (Level 0) in a location that gives read and execute permissions to all Canopy Python users, such as /usr/local/CanopyCore/ in this example.

2) **Do not run the Canopy application yet**, as this would lock in (though reversibly) an incompatible configuration. (If you did run the application prematurely, delete the Enthought subdirectory from your hidden directory "~/canopy" and proceed.

3) From within the Canopy installation directory (in this example, /usr/local/CanopyCore/), run the system-wide installation (common install):

    ./canopy_cli --common-install --install-dir=/usr/local/

This step creates common (system-wide) System (Canopy GUI app) and User Python environments in a Canopy_64bit subdirectory of the specified install-dir. These environments will be used and updated directly by the admin; all other users will subsequently inherit from these environments but will not be able to update them.

The directory structure on the central machine in this example would be:

    /usr/local/
    |  `- CanopyCore   # Level 0
    |      `- Canopy_64bit # Level 1
    |          |  `- System
    |          |      `- User

4) After the common System and User environments are created, the final dialog will ask "Do you want to make Canopy your default Python environment?" Explicitly select "No".

5) The first time that each user (other than the admin) starts Canopy from the Core installation (i.e., /usr/local/CanopyCore/canopy), their Python environments will automatically be created, inheriting from the admin’s common install. By default, their Python environments will be placed in their home directories (~/Enthought/Canopy_64bit/System/ and User/); the User environment is where their Python code will run (inheriting from the common User environment, which in turn inherits from CanopyCore).

**Result**

Showing the resulting inheritance levels schematically, you would have:

- Core Python (on Windows: Program Files) (level 0)
  - Common System environment for the Canopy app (level 1), also used directly by admin user
    - System environment for the Canopy app for user 1 (level 2)
    - System environment for the Canopy app for user 2 (level 2)
In contrast, if you had not run the common install step, you would have the following inheritance pattern:

Core Python (on Windows: Program Files) (level 0)
  System environment for the Canopy app for admin user (level 1)
  User environment for running user Python code for admin user (level 1)
  System environment for the Canopy app for user 1 (level 1)
  User environment for running user Python code for user 1 (level 1)
  System environment for the Canopy app for user 2 (level 1)  User environment for running user Python code for user 2 (level 1)  ....

Discussion

As you can see, the disadvantage of a common install is that it is more complex, introducing an additional level of virtual environment. While this does not significantly impact performance (just adds one more entry on sys.path), it requires more time to understand.

There are two main advantages to a common install, when many users share a system (whether local or networked):

a) The individual users' level 2 environments are normally very light weight, so there is normally no need to worry about their footprints.

b) When the admin updates her own Canopy GUI application (Common System environment) via Canopy update, or her own Python packages (Common User environment) via the Package Manager or enpkg utility, then all other users inherit those updates, saving work and providing a consistent base environment among users. An individual user can also update/install packages if desired, to override those same packages from the common environment.

Individual and Common updates

As just described, each user can, by default, update packages in their own Level 2 environment. If the administrator wishes to prevent this, the user’s level 2 User environment can be set to a directory to which the user has no write access.

As mentioned in Procedure step 3 above, the admin who set up the common install always runs in level 1 environments, directly using the Common System and User environments, so any
updates that she performs (whether of the Canopy GUI application or of packages) are inherited by all other users (running in level 2 environments). This is implemented by having the admin’s own locations.cfg file (Windows, found here: %APPDATA%\Enthought\Canopy\; in Linux: /usr/local/CanopyCore/) point to the common level 1 environment, whereas all other users’ locations.cfg files point to their own level 2 environments.

If the admin user does package management (e.g. with the "enpkg" utility) from a Canopy Command Prompt, she may again need to open that window with escalated admin privileges.