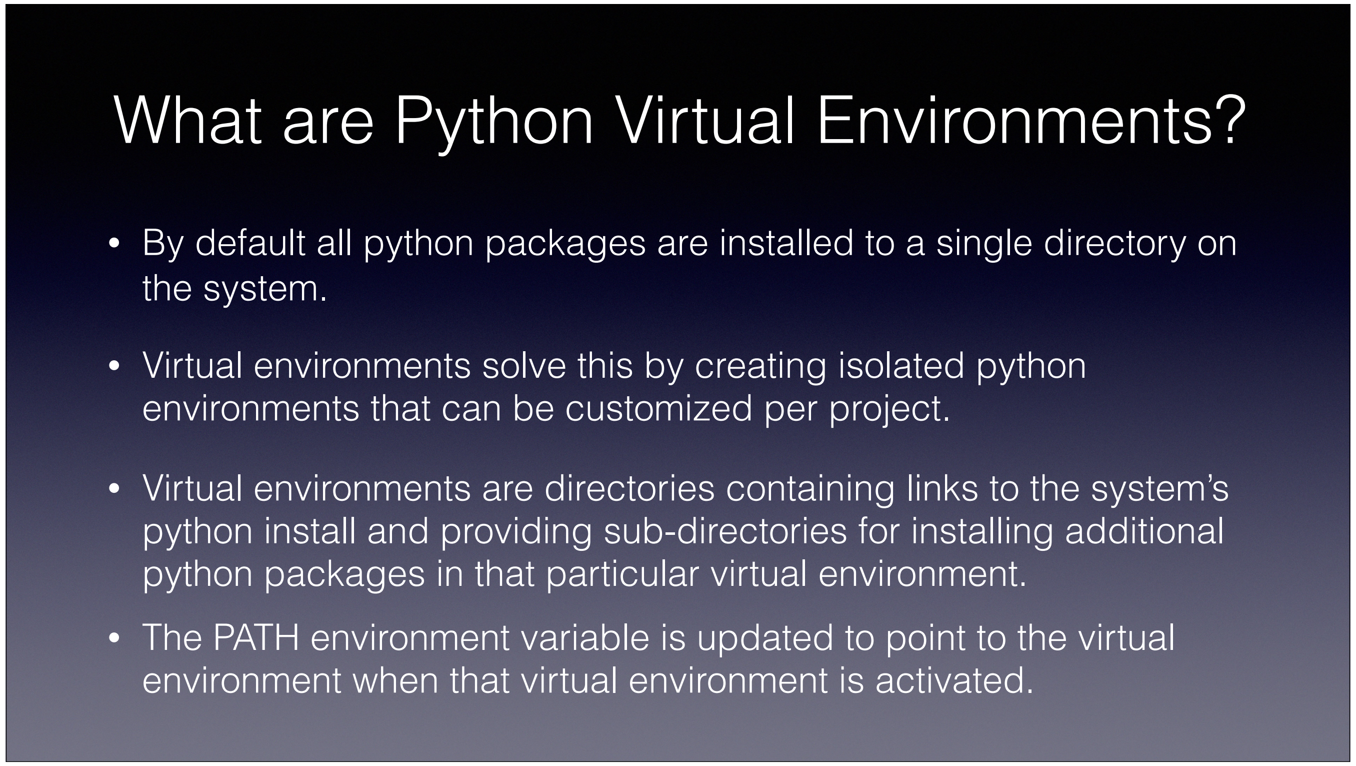
Overview of Python Virtual Environments

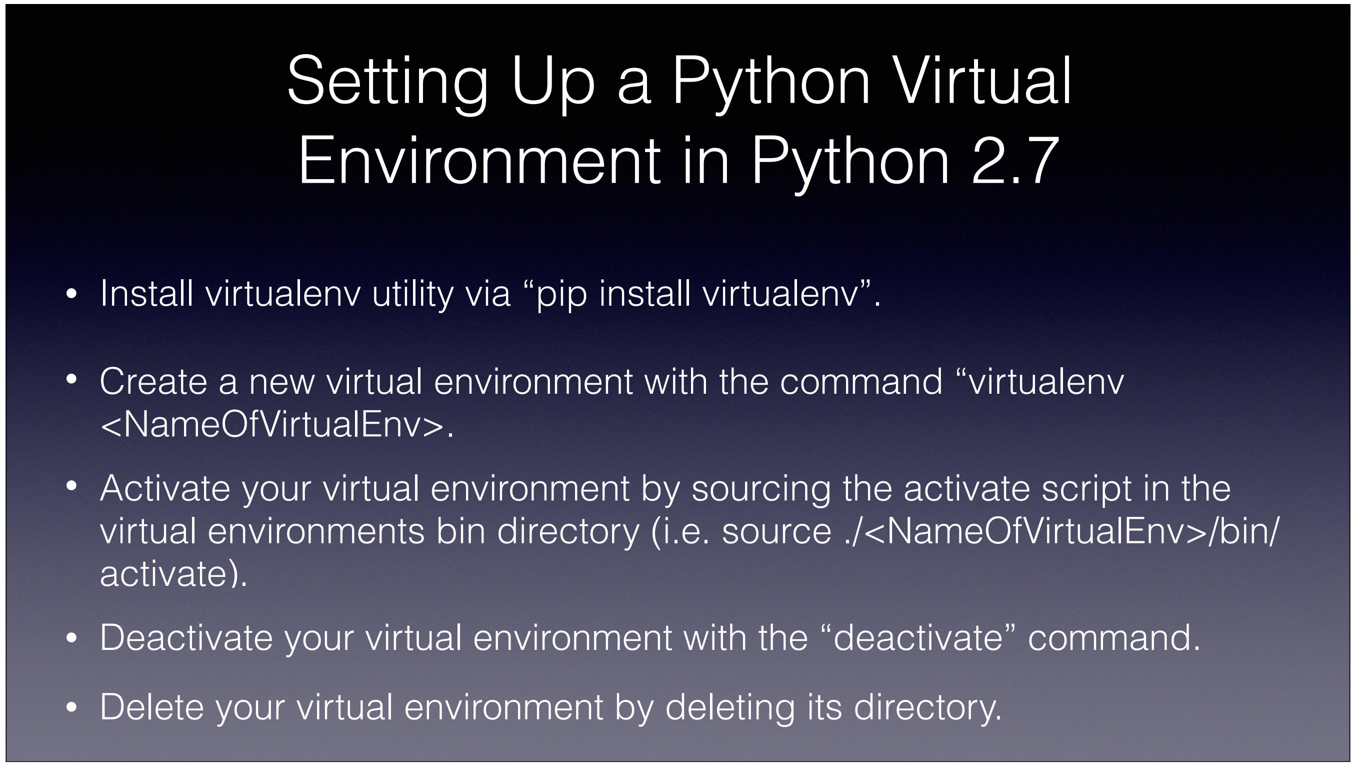
In this lecture I’m going to talk about Python virtual environments, why they’re important, and how to set them up and use them in Python 2.7 and Python 3.



What are Python Virtual Environments?

- By default all python packages are installed to a single directory on the system.
- Virtual environments solve this by creating isolated python environments that can be customized per project.
- Virtual environments are directories containing links to the system's python install and providing sub-directories for installing additional python packages in that particular virtual environment.
- The PATH environment variable is updated to point to the virtual environment when that virtual environment is activated.

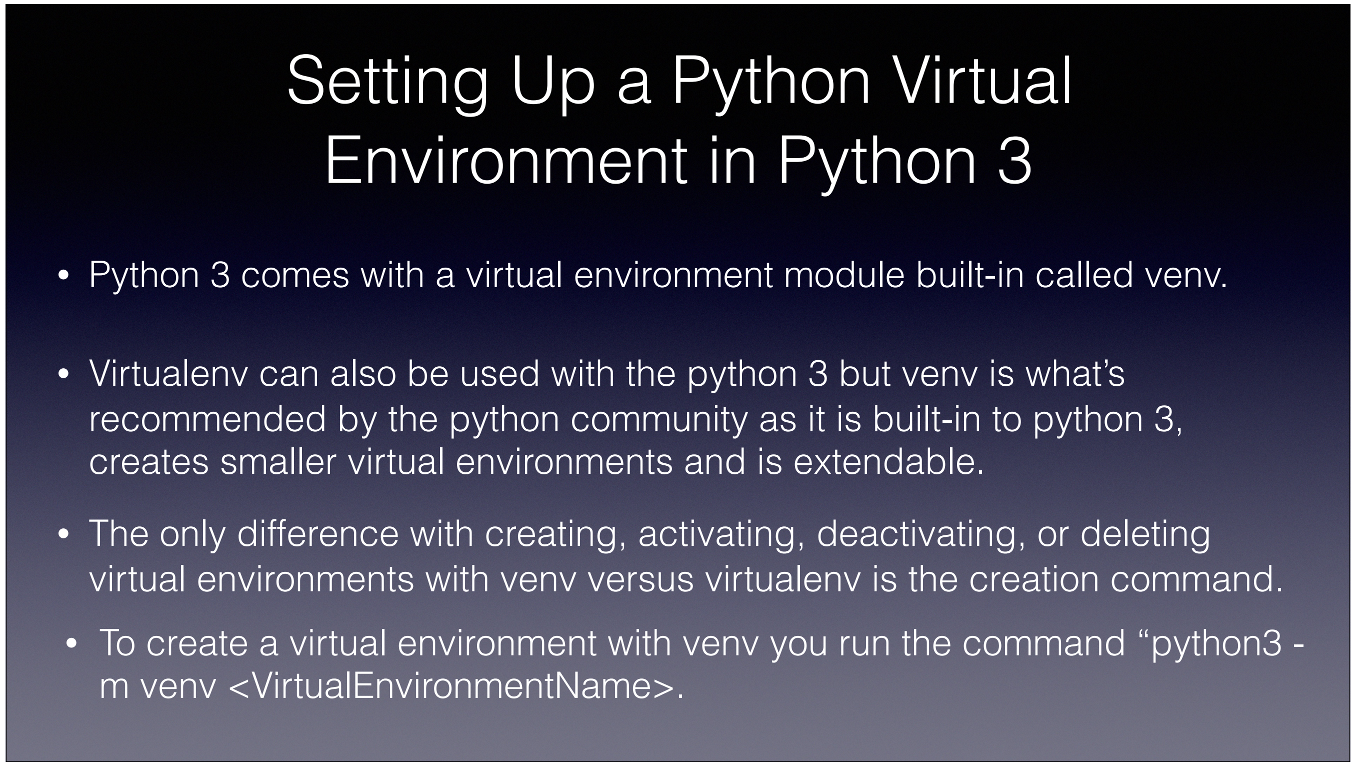
- So what are Python Virtual Environments?
- By default all python packages are installed to a single directory on the system which can become a problem when you have multiple python projects that have different and potentially conflicting dependencies. For example, project A needs version 1 of a particular library while project B needs version 2 of a particular library. This is a problem as the python runtime cannot have both versions of the library installed at the same time.
- Virtual environments solve this problem by creating isolated python environments that can be customized per project.
- This is done by creating a new directory for each virtual environment with links or copies of the python executable, library, and tools. Then sub-directories are added to hold the installed packages for that particular virtual environment.
- When a virtual environment is activated the PATH environment variable is updated to point first to the virtual environment's bin directory. This means that the virtual environment's python instance will be found and executed when python is run and sub-sequently the packages installed in that virtual environment will be used.



Setting Up a Python Virtual Environment in Python 2.7

- Install virtualenv utility via “pip install virtualenv”.
- Create a new virtual environment with the command “virtualenv <NameOfVirtualEnv>.
- Activate your virtual environment by sourcing the activate script in the virtual environments bin directory (i.e. source ./<NameOfVirtualEnv>/bin/activate).
- Deactivate your virtual environment with the “deactivate” command.
- Delete your virtual environment by deleting its directory.

- To use virtual environments with python 2.7 you need to install the virtualenv package with the command “pip install virtualenv”.
- Once virtualenv is installed you create a new virtual environment with the command “virtualenv” followed by the name of the new virtual environment.
- Once your virtual environment has been created you can activate it by sourcing the activate script in the environment’s bin directory.
- You can deactivate the virtual environment by simply typing the “deactivate” command.
- You can delete the virtual environment by deleting the top level directory that was created when you initially ran the virtualenv command.
- Now lets look at a quick example.



Setting Up a Python Virtual Environment in Python 3

- Python 3 comes with a virtual environment module built-in called venv.
- Virtualenv can also be used with the python 3 but venv is what's recommended by the python community as it is built-in to python 3, creates smaller virtual environments and is extendable.
- The only difference with creating, activating, deactivating, or deleting virtual environments with venv versus virtualenv is the creation command.
- To create a virtual environment with venv you run the command "python3 -m venv <VirtualEnvironmentName>.

- Python 3 comes with a built in module for creating virtual environments called venv.
- The Virtualenv command that we used with Python 2.7 can be used with Python 3 as well but using venv is the preferred approach as it's already built in to Python 3. It also creates smaller virtual environments and is extendable with the ability to use additional plugins.
- The commands to activate, deactivate, or delete a virtual environment are all the same for venv in Python 3 as they were for virtualenv with Python 2.7.
- The only difference is the command to create a virtual environment. The command for venv is "python3 -m venv" followed by the name of the new virtual environment.
- Now lets look at a quick example for python3 with venv.