APS106

UNIVERSITY OF TORONTO

How Does That Work?

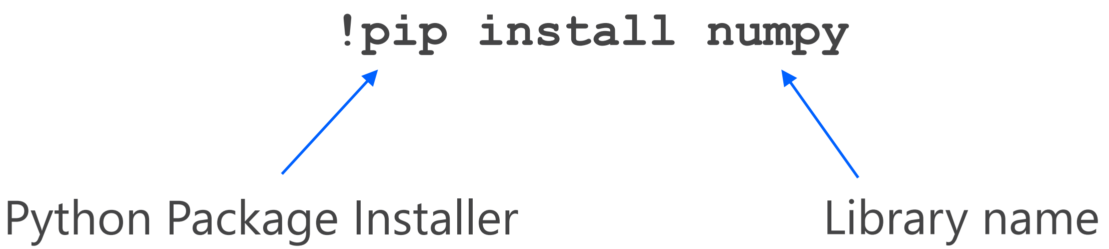
Python Environments and Packages/External Libraries

Why use external libraries?

- Using external libraries in Python can enhance your Python projects and make development more efficient
  - External libraries often provide functionalities that you would otherwise need to code from scratch
  - Many of these libraries are maintained by experts
  - In addition, a lot of libraries have advanced functionalities that are optimized for performance, let you work with complex data and do advanced processing tasks (like computer vision and machine learning!)

What are packages?

- Two main ways to get these:
  - Installation by running pip in terminal or a notebook cell

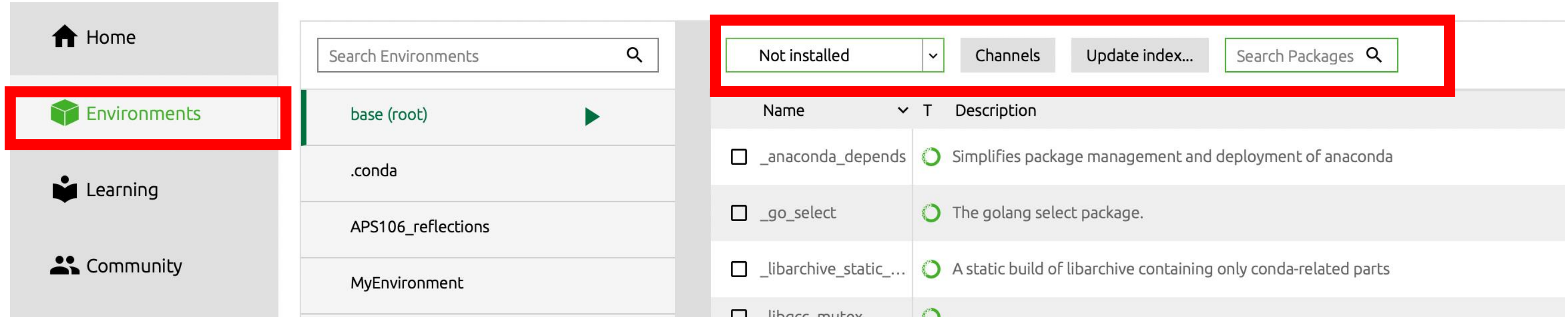


```
!pip install numpy
```

Python Package Installer

Library name

  - Environment management tools, like Anaconda



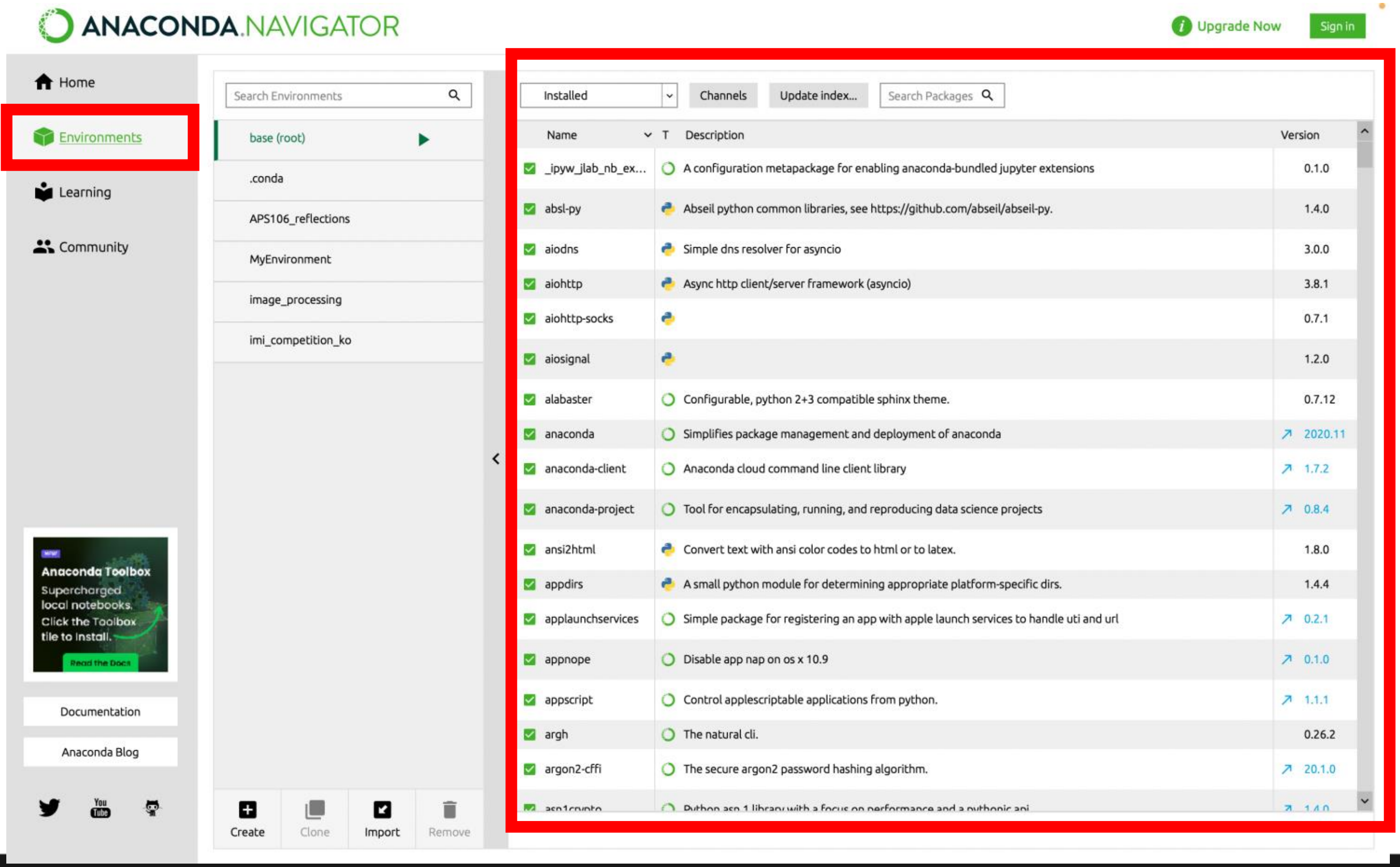
	Name	T	Description
□	_anaconda_depends	○	Simplifies package management and deployment of anaconda
□	_go_select	○	The golang select package.
□	_libarchive_static_...	○	A static build of libarchive containing only conda-related parts
□	_libgcc_mutex	○	

What is an Environment

- An environment is a directory that contains a specific collection of Python packages that have been installed, along with the specific version of Python.
- There are a few use cases to use them:
  - Need to ensure that dependencies are managed on large projects between different people.
  - Working with a specific version of code meant to run with specific libraries.
  - Resolving conflicts in different environment versions.

Anaconda simplifies environment management

- You can create, export, list, remove and update environments
- Also can create requirements.txt files from an environment, that can be shared as a pip install package as well



	Name	T	Description	Version
☑	_ipyw_jlab_nb_ex...	○	A configuration metapackage for enabling anaconda-bundled jupyter extensions	0.1.0
☑	absl-py		Abseil python common libraries, see https://github.com/abseil/abseil-py.	1.4.0
☑	aiodns		Simple dns resolver for asyncio	3.0.0
☑	aiohttp		Async http client/server framework (asyncio)	3.8.1
☑	aiohttp-socks			0.7.1
☑	aiosignal			1.2.0
☑	alabaster	○	Configurable, python 2+3 compatible sphinx theme.	0.7.12
☑	anaconda	○	Simplifies package management and deployment of anaconda	↗ 2020.11
☑	anaconda-client	○	Anaconda cloud command line client library	↗ 1.7.2
☑	anaconda-project	○	Tool for encapsulating, running, and reproducing data science projects	↗ 0.8.4
☑	ansi2html		Convert text with ansi color codes to html or to latex.	1.8.0
☑	appdirs		A small python module for determining appropriate platform-specific dirs.	1.4.4
☑	applaunchservices	○	Simple package for registering an app with apple launch services to handle uti and url	↗ 0.2.1
☑	appnope	○	Disable app nap on os x 10.9	↗ 0.1.0
☑	appscript	○	Control applescriptable applications from python.	↗ 1.1.1
☑	argh	○	The natural cli.	0.26.2
☑	argon2-cffi	○	The secure argon2 password hashing algorithm.	↗ 20.1.0
☑	asn1crypto	○	Python asn.1 library with a focus on performance and a pythonic api	↗ 1.4.0

Demo