Python Fundamentals

Shipping Working and Maintainable Code

Austin Bingham
@austin_bingham
austin@sixty-north.com

Presenter

Robert Smallshire
@robsmallshire
rob@sixty-north.com

pluralsight
hardcore developer training

unittest

unittest

unittest

automated
&
repeatable

TestCase

groups together related test functions

TestCase

groups together related test functions

Basic unit of test organization in unittest.

fixtures

code run before and/or after each test function

python™

fixtures

code run before and/or after each test function

```
def  test_line_count(self):
          "Check that the line count is correct."
          self.assertEqual(
     analyze_text(self.filename)[0], 4)
```

python™

fixtures

code run before and/or after each test function

set-up fixture

```
def  test_line_count(self):
          "Check that the line count is correct."
          self.assertEqual(
     analyze_text(self.filename)[0], 4)
```

python™

fixtures

code run before and/or after each test function

set-up fixture

```
def  test_line_count(self):
     "Check that the line count is correct."
     self.assertEqual(
          analyze_text(self.filename)[0], 4)
```

tear-down/clean-up fixture

assertions

specific tests for conditions and behaviors

assertions

specific tests for conditions and behaviors

x.is_valid()

assertions

specific tests for conditions and behaviors

`x.is_valid()`

`x == y`

assertions

specific tests for conditions and behaviors

x.is_valid()

x == y

raise ValueError()

assertions

specific tests for conditions and behaviors

x.is_valid()

x == y

raise ValueError()

If an assertion fails, then a test fails.

Test-Driven Development

Test-Driven Development

take filename argument

Test-Driven Development

take filename argument

read file

Test-Driven Development

take filename argument

read file

calculate lines and characters

python™

PDB

The Python DeBugger

PDB

programmatic access

```
>>> import pdb
>>> pdb.set_trace()
```

is_palindrome()

determines if an integer is a palindrome or not

is_palindrome()

determines if an integer is a palindrome or not

12321

is_palindrome()

determines if an integer is a palindrome or not

12321
2468642

is_palindrome()

determines if an integer is a palindrome or not

12321
2468642
11235813

is_palindrome()

determines if an integer is a palindrome or not

12321

2468642

~~11235813~~

virtual environment

light-weight, self-contained Python installation

on Windows:

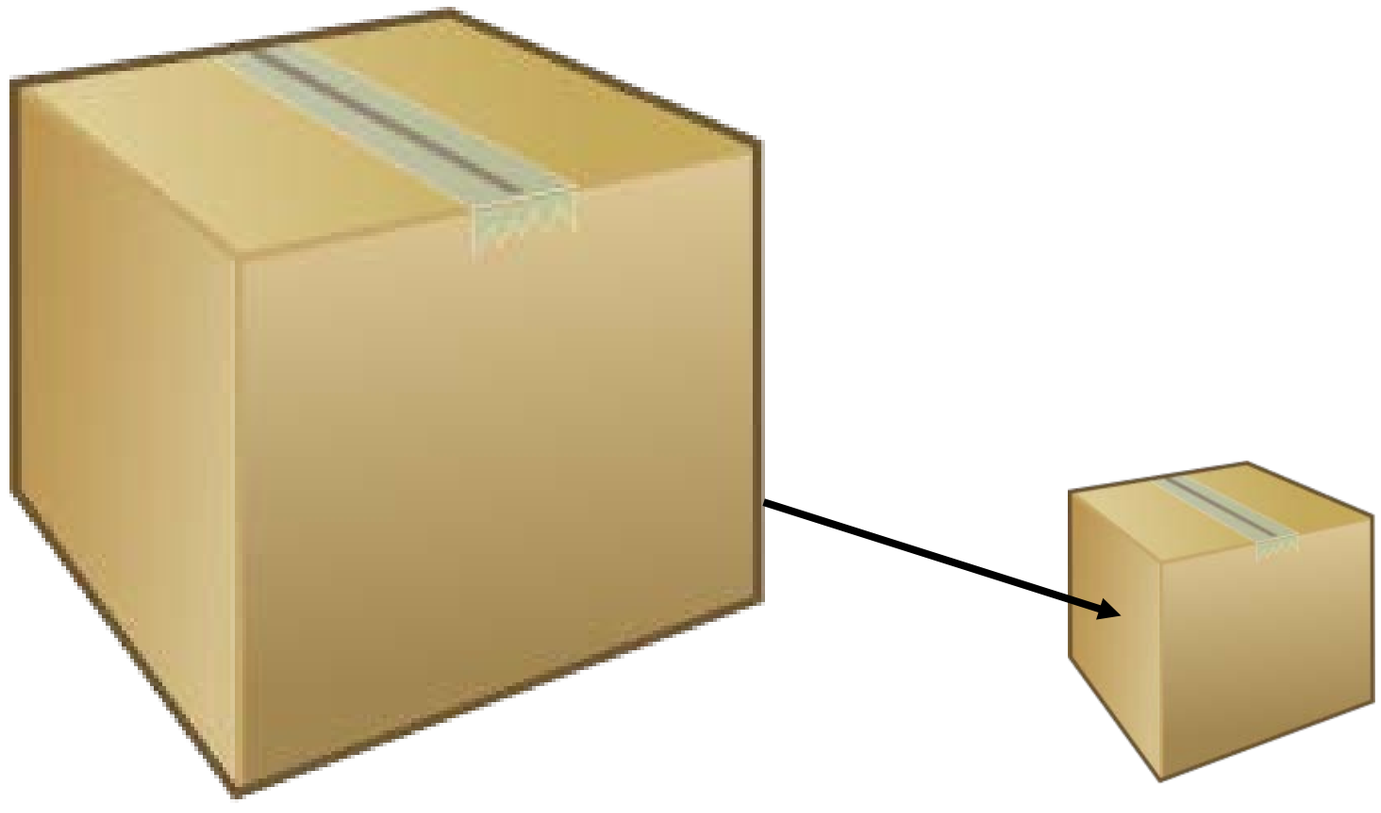
```
> venv3\bin\activate
```

python™

packaging

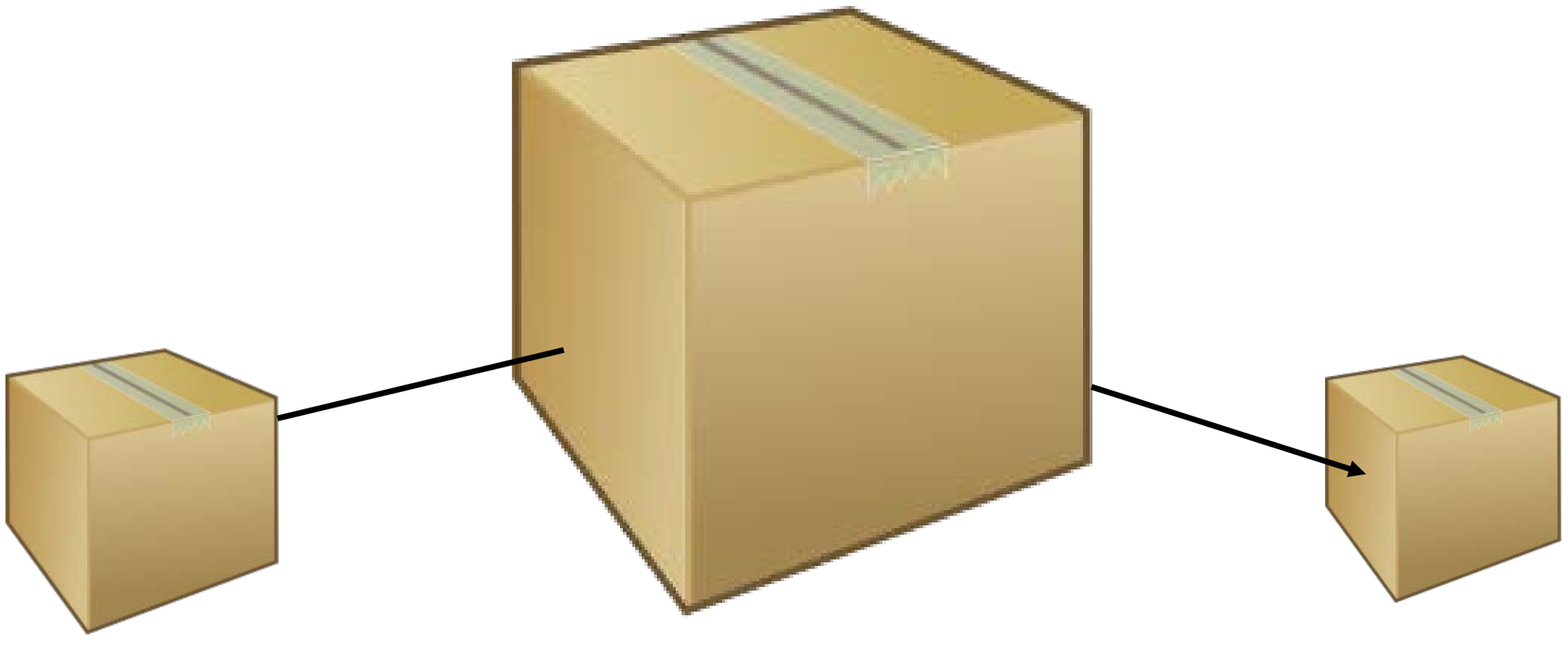
python™

packaging



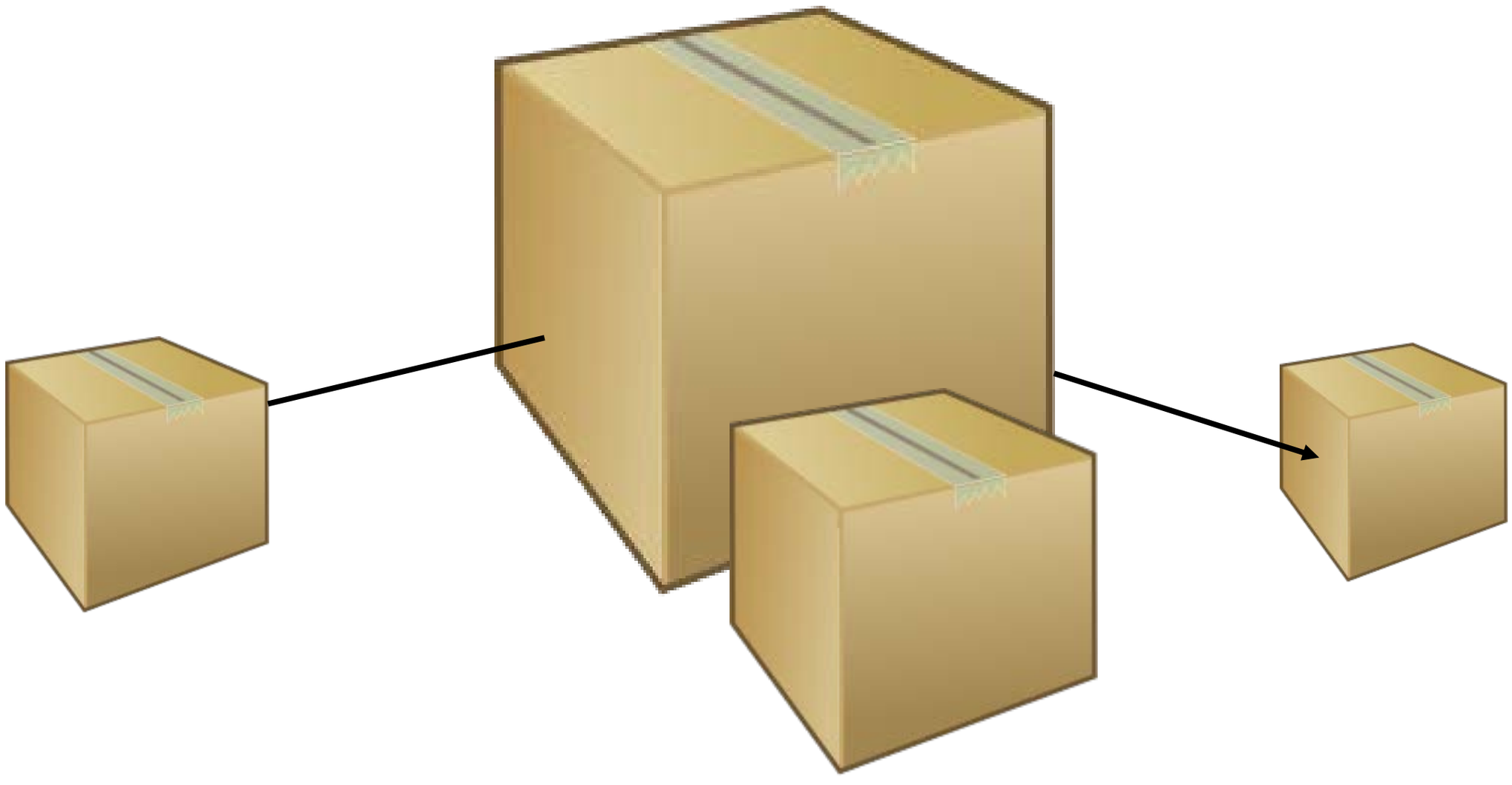
python™

packaging



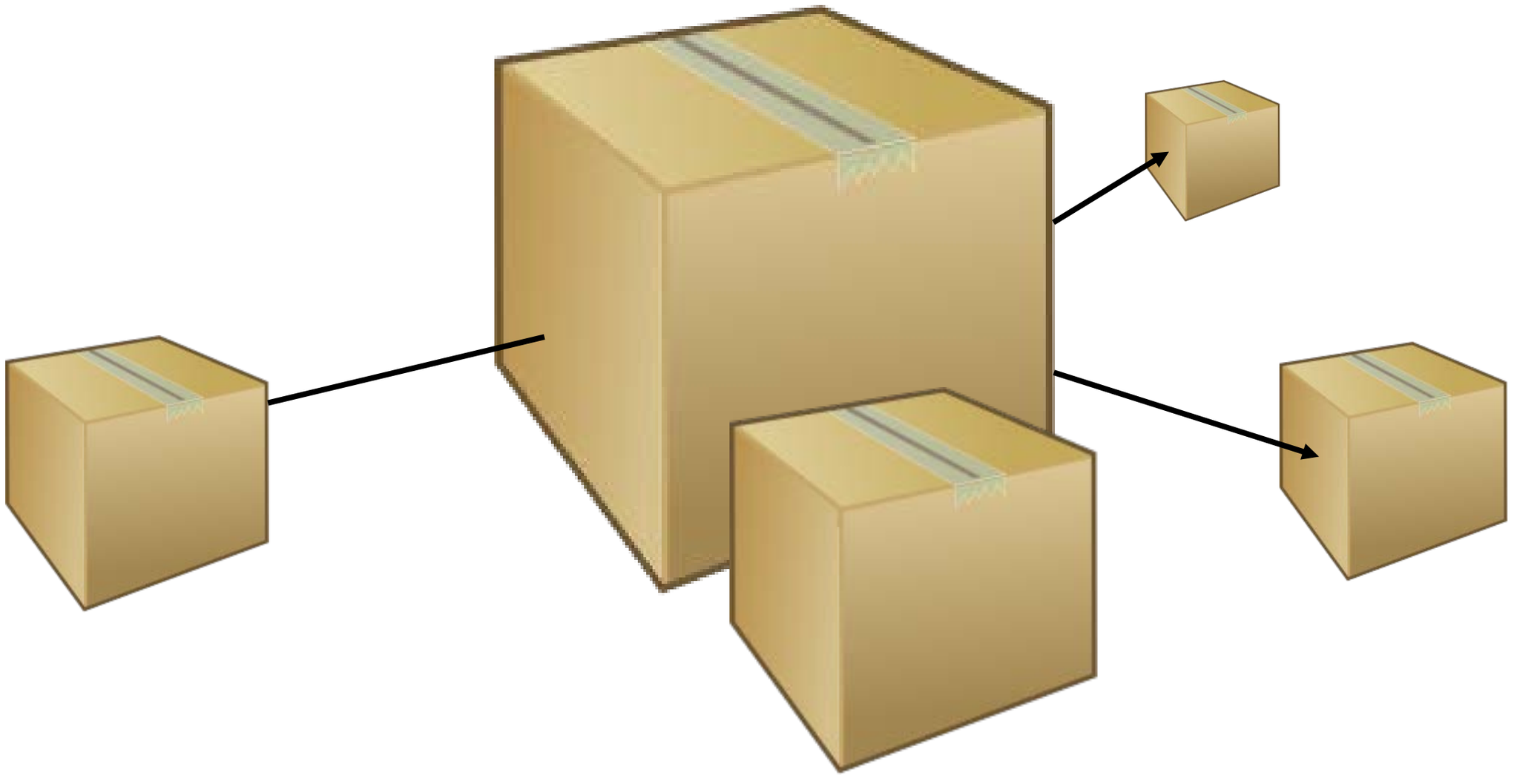
python™

packaging



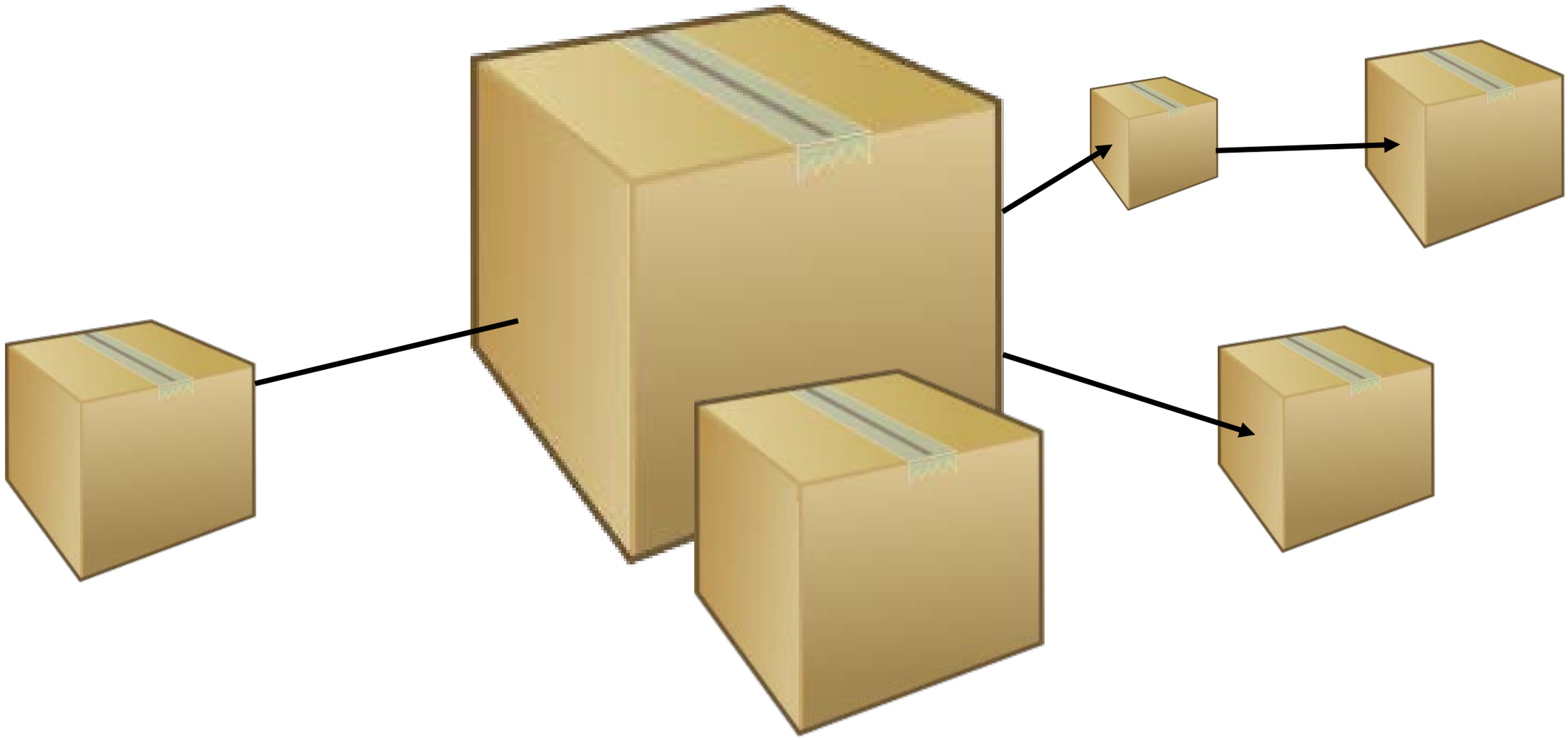
python™

packaging



python™

packaging



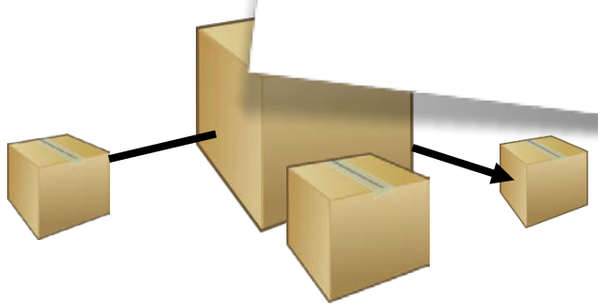
python™

packaging

```
from distutils.core import setup

setup(
  name = 'palindrome',
  version = '1.0',
  py_modules  = ['palindrome'],

  # metadata
  author = 'Austin Bingham',
  author_email = 'austin@sixty-north.com',
  description = 'A module for finding palindromic integers.',
  license = 'Public domain',
  keywords = 'example',
)
```

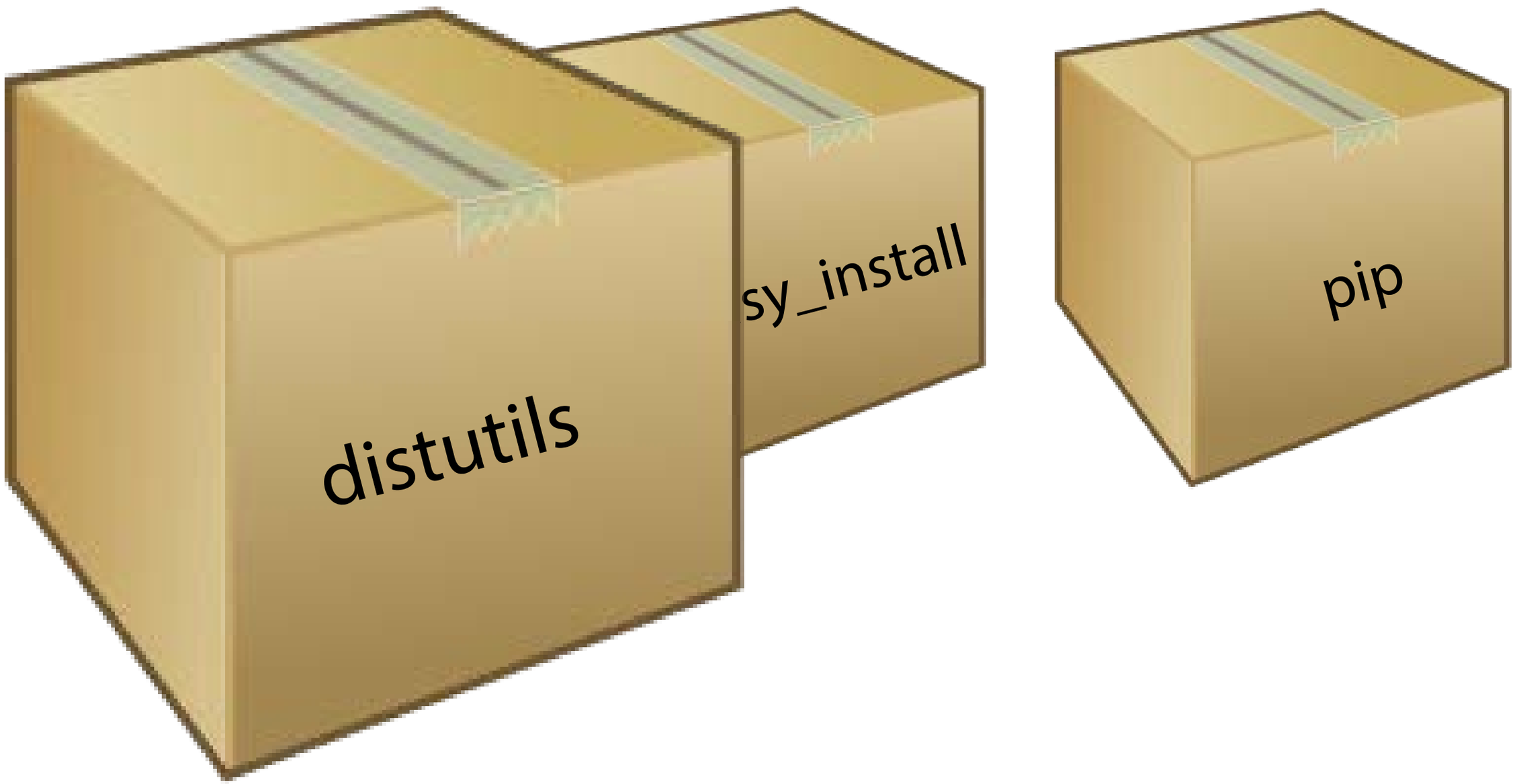


python™

installing

python™

installing

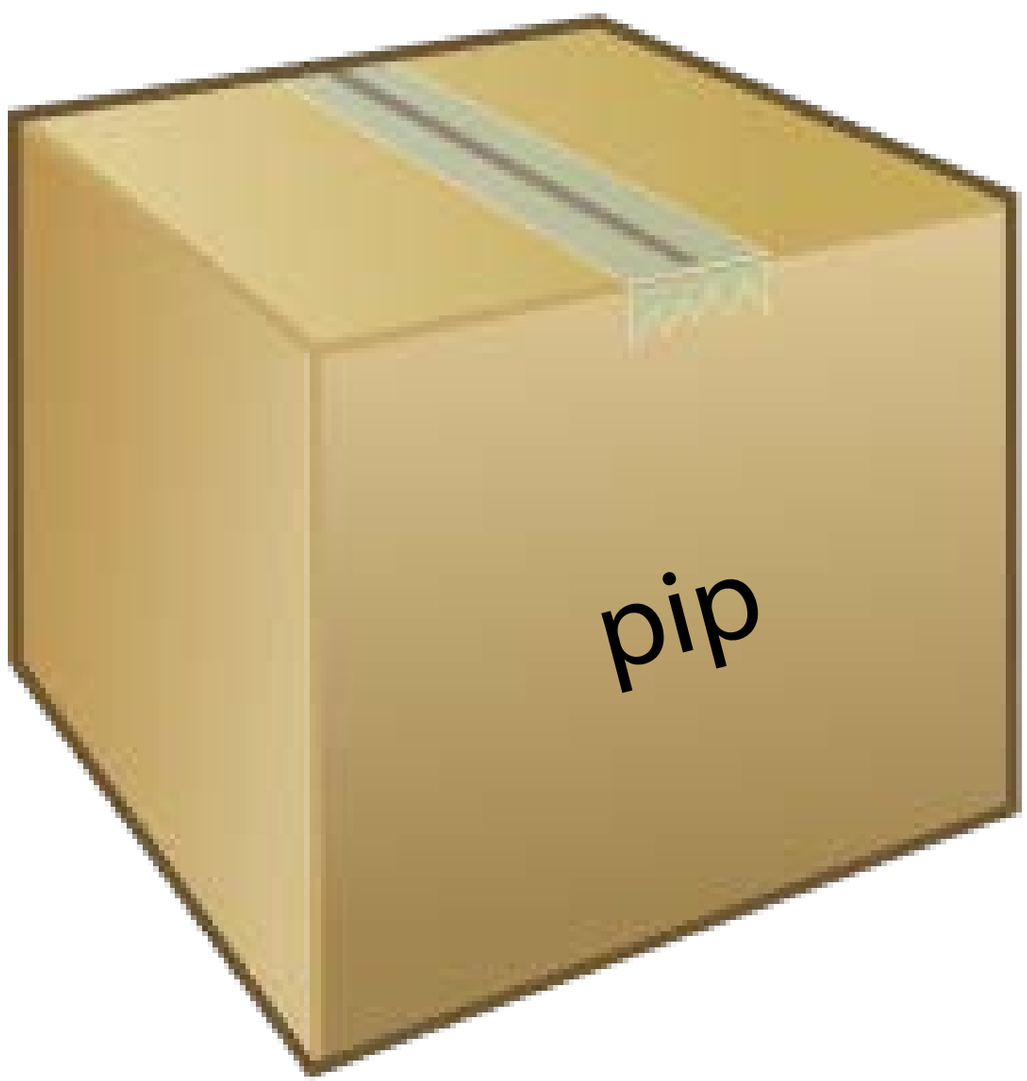


python™

installing

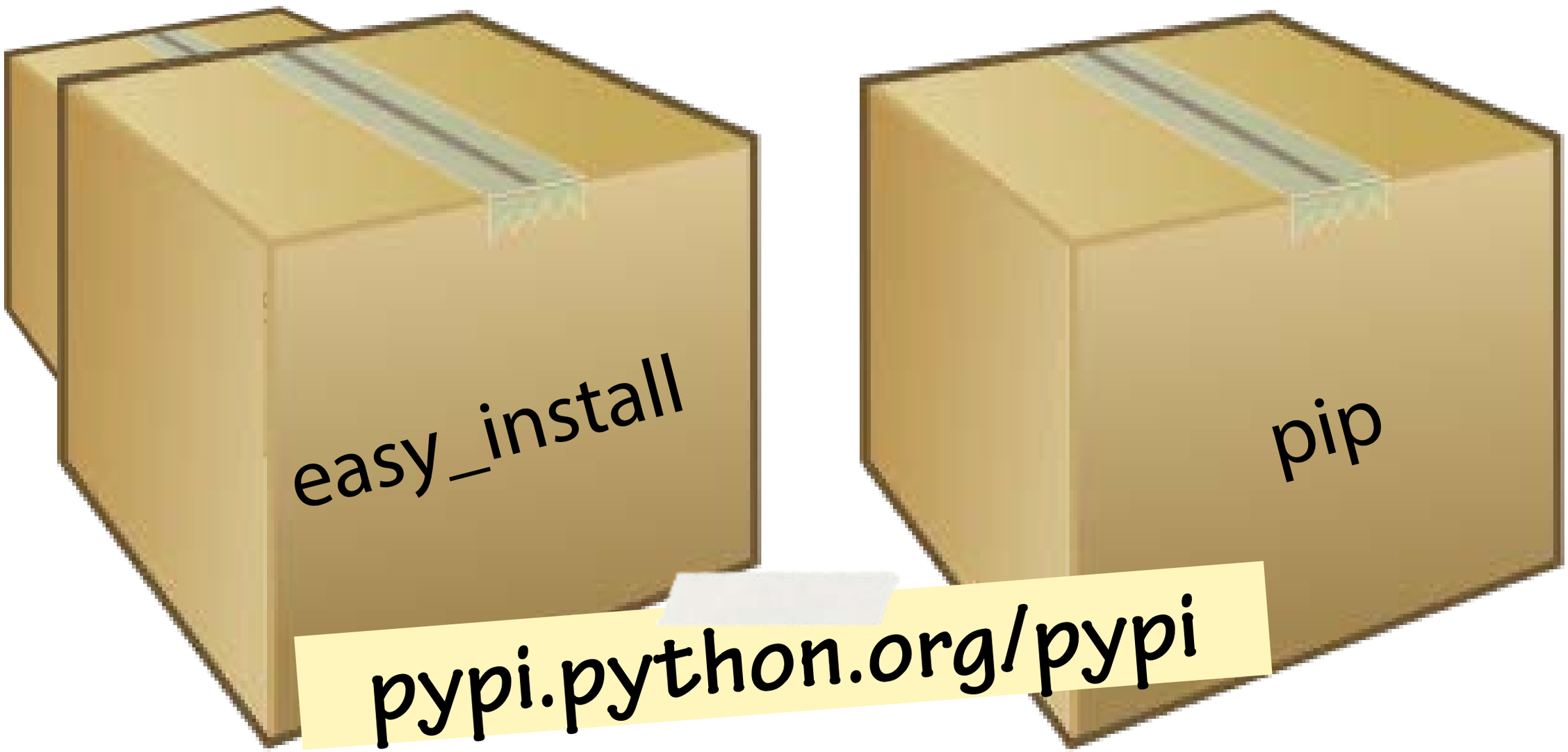
python™

installing



python™

installing



python™

installing

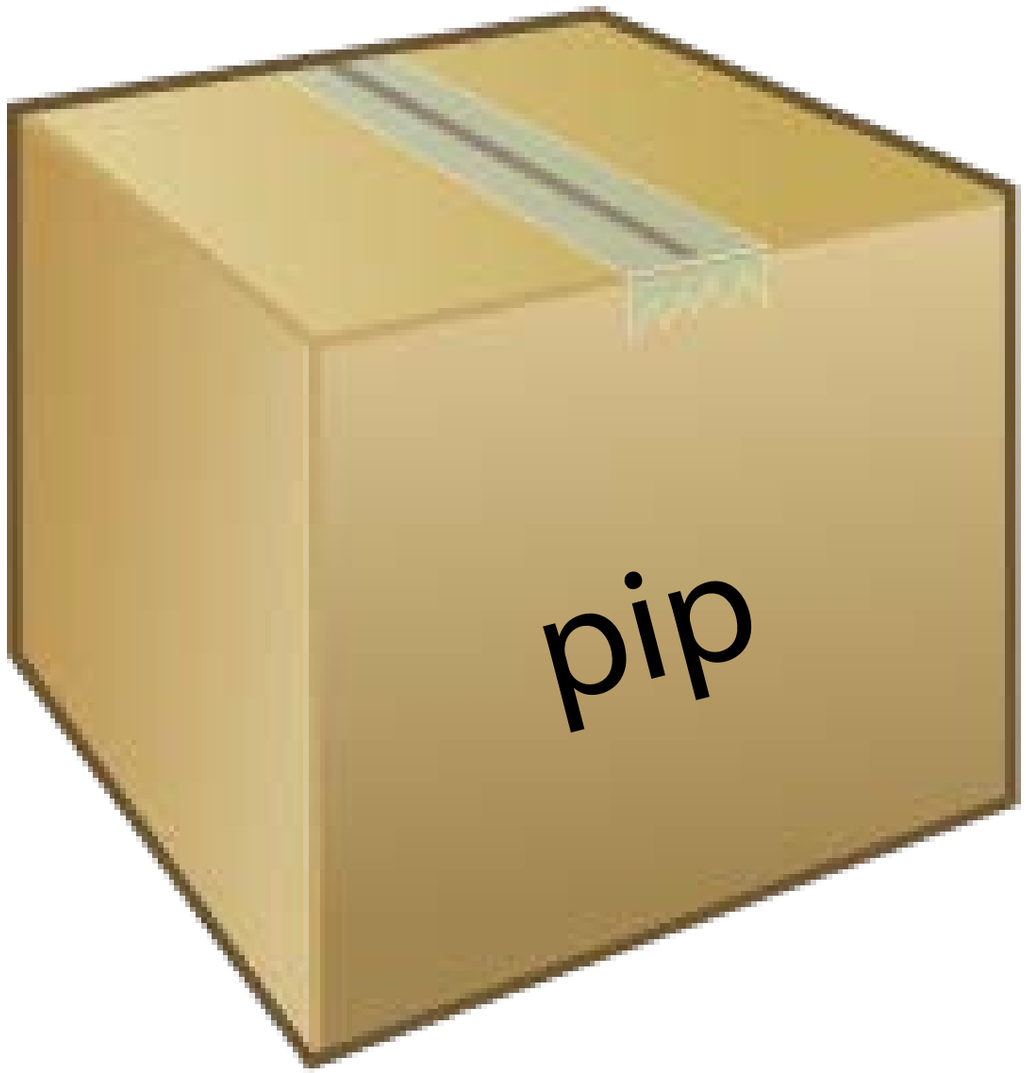
python™

installing

```
$ easy_install <package name>
```

python™

installing



python™

installing

installing

```
$ pip install <package name>
```

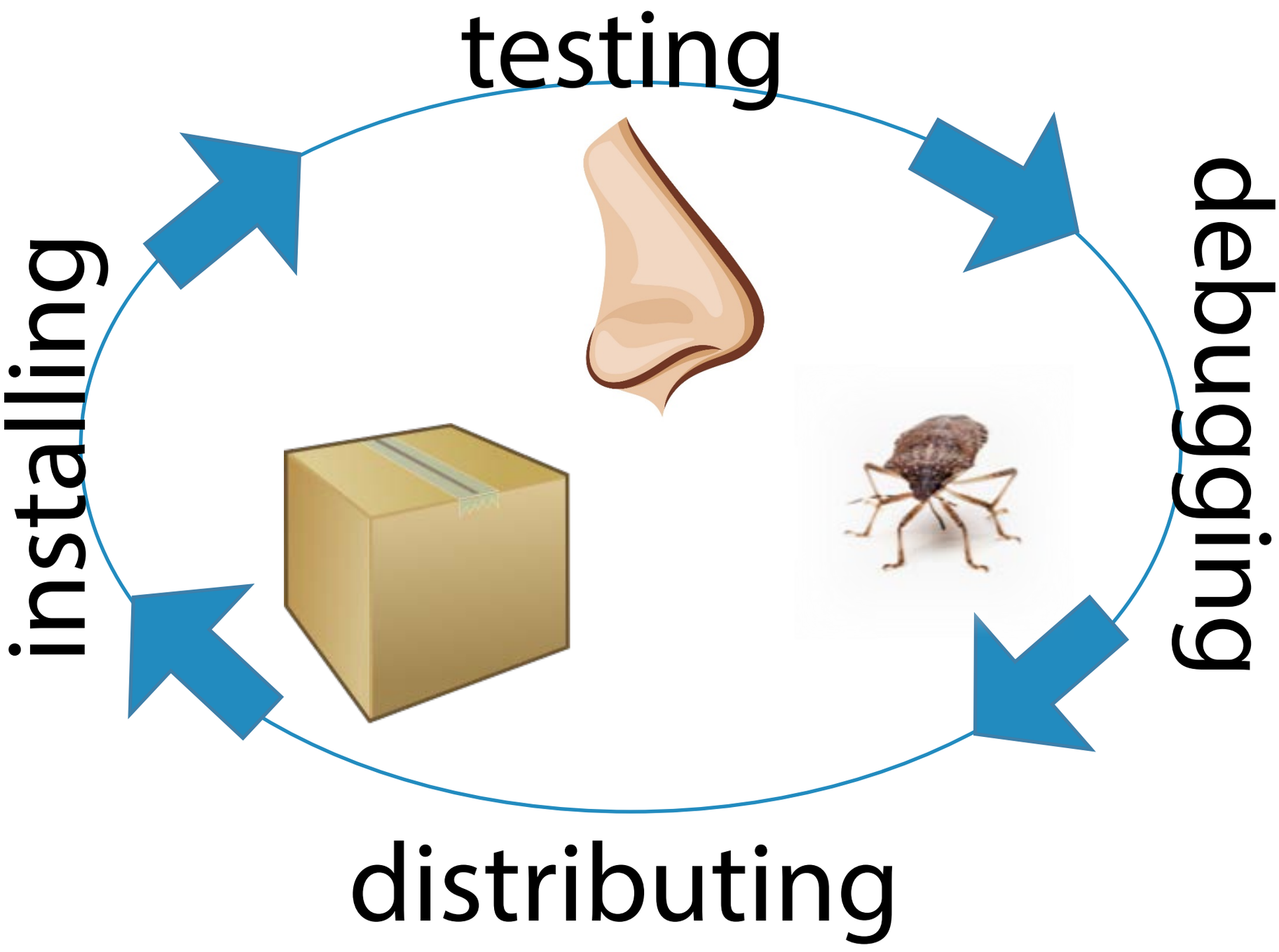
Moment of Zen

In the face of ambiguity, refuse the temptation to guess.

To guess is to know
That you have left something out.
What are you missing?

python

Shipping working and maintainable code
Summary



Shipping working and maintainable code
Summary

- unittest is a framework for developing reliable automated tests
- You define test cases by subclassing from unittest.TestCase
- unittest.main() is useful for running all of the tests in a module
- setUp() and tearDown() run code before and after each test method
- Test methods are defined by creating method names that start with test
- TestCase.assert... methods make a test method fail when the right conditions aren't met
- Use TestCase.assertRaises() in a with-statement to check that the right exceptions are thrown in a test
- Python's standard debugger is called PDB
- PDB is a standard command-line debugger
- pdb.set_trace() can be used to stop program execution and enter the debugger
- Your REPL's prompt will change to (Pdb) when you're in the debugger

Shipping working and maintainable code
Summary

- You can access PDB's built-in help system by typing help
- Use “python -m pdb <script name>" to run a program under PDB from the start
- PDB's where command shows the current call stack
- PDB's next command lets execution continue to the next line of code
- PDB's continue command lets program execution continue indefinitely, or until you stop it with control-c
- PDB's list command shows you the source code at your current location
- PDB's return command resumes execution until the end of the current function
- PDB's print command lets you see the values of objects in the debugger
- Use quit to exit PDB
- Virtual environments are light-weight, self-contained Python installations that any user can create

Shipping working and maintainable code Summary

- pyvenv accepts both a source-installation argument as well as a directory name into which it creates the new environment
- To use a virtual environment, you need to run its activate script
- When you activate a virtual environment, your prompt is modified to remind you
- The distutils package is used to help you distribute your Python code
- distutils is generally used inside a setup.py script which users run to install your software
- The main function in distutils is setup()
- setup() takes a number of arguments describing both the source files as well as metadata for the code
- The most common way to use setup.py is to install code using python setup.py install
- setup.py can also be used to generate distributions of your code
- Distributions can be zip files, tarballs, or several other formats

Shipping working and maintainable code Summary

- Pass --help to setup.py to see all of its options
- Three common tools for installing third-party software are distutils, easy_install, and pip
- The central repository for Python packages is the Python Package Index, also called PyPI or "cheeseshop"
- You can install easy_install by downloading and running distribute_setup.py
- You use easy_install to install modules by running easy_install package-name from the command line
- You can install pip via easy_install
- To install modules with pip, use the subcommand notation pip install package-name

Shipping working and maintainable code Summary

- divmod() calculates the quotient and remainder for a division operation at one time
- reversed() function can reverse a sequence
- You can pass -m to your Python command to have it run a module as a script
- Debugging makes it clear that Python is evaluating everything at run time
- You can use the __file__ attribute on a module to find out where its source file is located
- Third-party python is generally installed into your installation's site-packages directory
- nose is a useful tool for working with unittest-based tests

Shipping working and maintainable code Summary

- divmod() calculates the quotient and remainder for a division operation at one timereversed() function can reverse a sequenceYou can pass -m to your Python command to have it run a module as a scriptDebugging makes is clear that Python is evaluating everything at run timeYou can use the __file__ attribute on a module to find out where it's source file is locatedThird-party python is generally installed into your installation's site-packages directorynose is a useful tool for working with unittest-based tests