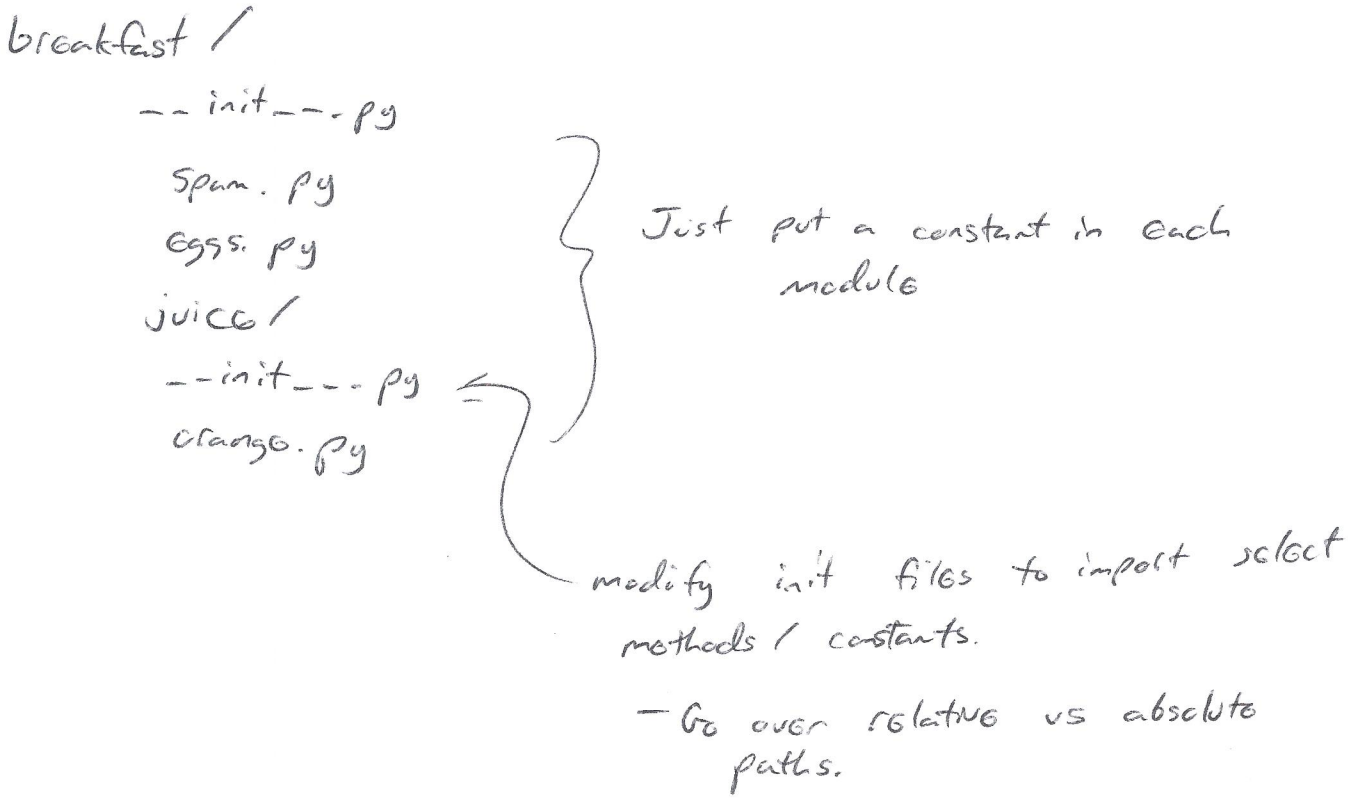
LESSON 8 PACKAGES, PIP, PyTest

Make a simple package structure



- Go over relative vs absolute paths.

- Try import breakfast.spam, note that breakfast is a bound name as well as breakfast.spam

Note that you can access the value SPAM

- Do some examples with breakfast.juice.orange
- Do examples only importing breakfast directly
- Note that a directory/package is really just a module defined by `__init__` with sub-modules potentially available.
- Package docstrings can be added to `__init__`.

Third-party packages and site-packages

In addition to the current directory, the standard lib, and PYTHONPATH, the interpreter will look for third-party packages in a site-packages directory.

By default, site-packages applies to everyone on the machine, and maybe even the OS itself!!!

POSIX → Use 'which python' and 'python --version' to see what python you are using!

Generally we don't want to add/upgrade 3rd party site-packages for the system python, or clutter up our system for testing libraries.

SOLUTION: Create a special 'virtual' environment with its own site-packages - install packages just in that environment. Discard environments when no longer needed.

venv module

↳ python -m venv myvenv

↑ creates virtual environment in specified directory

- myvenv/bin/activate ← Sets current shell to use this environment

deactivate ← ~~restores~~ stops using environment

rm -r myvenv ← Just delete to clean up

pip — tool for installing / managing third-party site packages from PyPA repository. (formerly known as cheese shop)

pip freeze ← see what is installed

pip install requests ← install a package ← Do Example

pip uninstall requests ← uninstall a package

↓

Play with requests a bit

UNIT TESTING!

unittest module ← standard library, java based

pytest ← 3rd party, but really nice

Test the smath.py module from last time → Also do a pytest.raises example for a type error with bad input

make a test_smath.py module.

↳ try testing $2.3^2 = 5.29$ with floating point error, then use approx.

Explain rewriting of 'assert'. → Do example of a normal assert statement in a module use '-O' switch

Pytest discovery → any module or function prefixed with 'test_'
any class prefixed with Test (and no init)

Now do examples of running specific tests, i.e.
pytest test_smath.py:: test_square_float, etc...