All About Importing Modules and from Packages

Module?

A module is just a Python file with the corresponding `.py` extension. So if you're talking about the math module then there is a corresponding `math.py` file that contains functions, classes, and constants that are meant to be used by other Python files.

Where does Python look for modules

If you have written a python module under say `a_module.py` in a directory of `code`.

And you have a script called `a_script.py` in directory called `scripts`. You would like to use the `a_module` in `a_script.py` by importing it.

```
import a_module
```

Then you try to run the `a_script.py` by running `python3 scripts/a_script.py` it will fail with

```
$ python3 scripts/a_script.py
Traceback (most recent call last):
  File "scripts/a_script.py", line 1, in <module>
    import a_module
ModuleNotFoundError: No module named 'a_module'
```

When Python imports a module it will try to find a package or module. But where does it look? Python has a simple algorithm for finding a module with a given name. It will look for a file called `a_module.py` in the directories listed in the variable `sys.path`.

```
>>> import sys
>>> type(sys.path)
<class 'list'>
>>> for path in sys.path:
```

```
...     print(path)
...
/Users/brettmz-admin/dev_trees/psych-214-fall-2016/sphinxext
/usr/local/Cellar/python/3.7.2_1/Frameworks/Python.framework/Versions/3.7/lib/python37.zip
/usr/local/Cellar/python/3.7.2_1/Frameworks/Python.framework/Versions/3.7/lib/python3.7
/usr/local/Cellar/python/3.7.2_1/Frameworks/Python.framework/Versions/3.7/lib/python3.7/lib-dynload
/Users/brettmz-admin/Library/Python/3.7/lib/python/site-packages
/Users/brettmz-admin/dev_trees/grin
/Users/brettmz-admin/dev_trees/rmdex
/usr/local/lib/python3.7/site-packages
```

It doesn't search recursively, it will only search in the directory that is listed under `sys.path`. And as you can see the `code` directory is not in `sys.path` which is why Python could not import `a_module.py`.

To fix this you can just simply append to the `sys.path` list like so:

```
import sys
sys.path.append('code')


import a_module
```

Now this will work as expected. This simple search algorithm for module also works for packages, it searches for packages then the module the same way.

> Information from: https://bic-berkeley.github.io/psych-214-fall-2016/sys_path.html

What is Namespace package and Regular Package

A namespace packages are special packages that allows you to unify two packages with the same name but are at different directories:

```
path1
+--namespace
   +--module1.py
   +--module2.py
path2
```

```
+--namespace
   +--module3.py
   +--module4.py
```

Notice that in order to make Namespace package work you would have to add the two paths `path1` and `path2` to `sys.path`. Then you can import the four modules by doing

```
from namespace import module1
from namespace import module2
from namespace import module3
from namespace import module4
```

Or import specific functions from the module

```
from namespace.module1 import boo
```

Namespace packages basically unifies the two packages with the same name in a single namespace. You can import all four modules freely.

However, if either one of the `namespace` packages gain an `__init__.py` then it will become a normal package, and the unification is no longer as the other directory will be ignored.

If both have `__init__.py` the first one encountered in `sys.path` is the one being used.

Regular package

A regular package is a collection of modules. In order to make Python recognize that it is a regular package you must add `__init__.py`. Without `__init__.py` then it will be interpreted as a namespace package which doesn't fit 99% of normal use cases.

So the moral of the story is that if you're creating packages, just put `__init__.py` in it unless you have the special use case of needing to unify two namespace packages that are in different directory.

Regular packages are also searched in `sys.path` just like everything else.

Revision #3
Created 2023-03-14 03:05:53 UTC by Tamarine
Updated 2023-03-14 18:58:07 UTC by Tamarine