A DEEP DIVE INTO PYTHON IMPORT HOOKS

Louis Taylor

PyCon UK – September 2015

@kragniz

PYTHON IMPORT HOOKS FOR FUN AND PROFIT

Louis Taylor

PyCon UK – September 2015

@kragniz

PYTHON IMPORT HOOKS FOR FUN BUT PROBABLY NOT TOO MUCH PROFIT

Louis Taylor

PyCon UK – September 2015

@kragniz

PYTHON IMPORT HOOKS FOR FUN

Louis Taylor

PyCon UK – September 2015

@kragniz

INTRODUCTION TO ME

- I’m an undergraduate at Aberystwyth university

INTRODUCTION TO ME

- I'm an undergraduate at Aberystwyth university
- Currently a core developer on OpenStack (Image Service)

INTRODUCTION TO ME

- I'm an undergraduate at Aberystwyth university
- Currently a core developer on OpenStack (Image Service)
- Chief snake wrangler for AberSailbot

INTRODUCTION TO ME

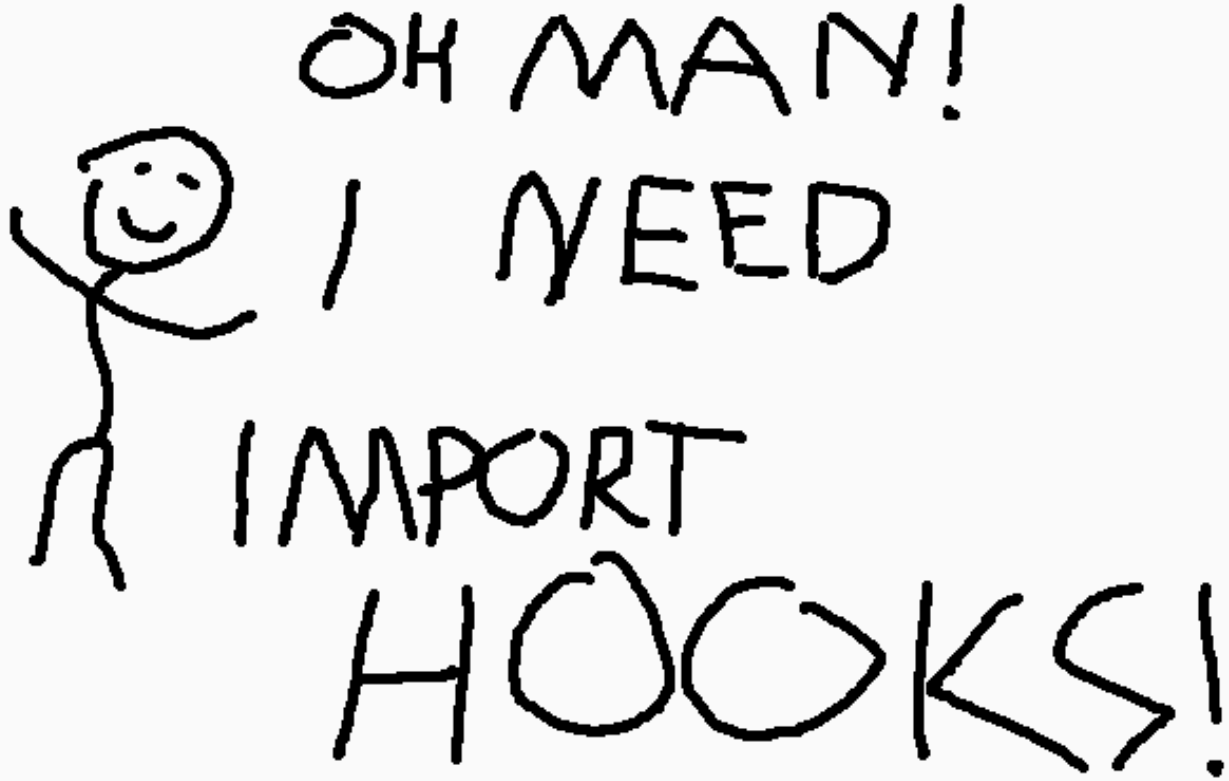
- I'm an undergraduate at Aberystwyth university
- Currently a core developer on OpenStack (Image Service)
- Chief snake wrangler for AberSailbot
- First talk at a pycon

INTRODUCTION TO ME

- I'm an undergraduate at Aberystwyth university
- Currently a core developer on OpenStack (Image Service)
- Chief snake wrangler for AberSailbot
- First talk at a pycon
- `[SWEATING INTENSIFIES]`

Why this talk?

01:33 <+Sakura> kragniz: it would be cool if you could make python think that json files were regular modules


OH MAN!
I NEED
IMPORT
HOOKS!

But who was import hook?

PYTHON IMPORT RECAP

```
import marmite
```

PYTHON IMPORT RECAP

```
import marmite
```

looks for a file named **"marmite.py"** in some directories

↓

???

↓

you get a python module

PYTHON IMPORT RECAP

1. search for module

PYTHON IMPORT RECAP

1. search for module
2. load module

PYTHON IMPORT RECAP

1. search for module
2. load module
3. bind module to some name

SEARCHING FOR A MODULE

1. look it up in `sys.modules`

SEARCHING FOR A MODULE

1. look it up in `sys.modules`
2. invoke the python import protocol

SEARCHING FOR A MODULE

1. look it up in `sys.modules`
2. invoke the python import protocol
3. raise `ImportError`

A SMALL DIGRESSION

This changed in python 3.4

Backwards compatible with python 2.7

PYTHON IMPORT PROTOCOL

What's that import protocol?

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders
2. Loaders

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders
2. Loaders

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders
2. Loaders

Defaults for build-in modules

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders
2. Loaders

Defaults for build-in modules, frozen modules

PYTHON IMPORT PROTOCOL

What's that import protocol?

Two objects:

1. Finders
2. Loaders

Defaults for build-in modules, frozen modules and modules on an import path

PYTHON IMPORT PROTOCOL - FINDERS

Finders locate modules, but don’t load them

PYTHON IMPORT PROTOCOL - FINDERS

Finders locate modules, but don't load them

```
class ExampleFinder(object):
    def find_module(fullname, path=None):
        some stuff
```

PYTHON IMPORT PROTOCOL - FINDERS

Finders locate modules, but don't load them

```
class ExampleFinder(object):
    def find_module(fullname, path=None):
        some stuff
```

The `find_module` method should either:

- Raise an exception

PYTHON IMPORT PROTOCOL - FINDERS

Finders locate modules, but don't load them

```
class ExampleFinder(object):
    def find_module(fullname, path=None):
        some stuff
```

The `find_module` method should either:

- Raise an exception
- Return None

PYTHON IMPORT PROTOCOL - FINDERS

Finders locate modules, but don't load them

```
class ExampleFinder(object):
    def find_module(fullname, path=None):
        some stuff
```

The `find_module` method should either:

- Raise an exception
- Return None
- Return a loader object

PYTHON IMPORT PROTOCOL - LOADERS

Loaders actually load modules

PYTHON IMPORT PROTOCOL - LOADERS

Loaders actually load modules

```
class ExampleLoader(object):
    def load_module(self, fullname):
```

PYTHON IMPORT PROTOCOL - LOADERS

Loaders actually load modules

```
class ExampleLoader(object):
    def load_module(self, fullname):
```

The `load_module` method needs to return a module object

PYTHON IMPORT PROTOCOL - LOADERS

Loaders actually load modules

```
class ExampleLoader(object):
    def load_module(self, fullname):
```

The `load_module` method needs to return a module object

Or raise an exception (probably `ImportError`)

ENABLING IMPORT HOOKS

How to make python know about these hooks?

ENABLING IMPORT HOOKS

How to make python know about these hooks?

```
import sys
sys.meta_path.append(HookObject)
```

WHAT CHANGED IN PYTHON 3.4

Both `find_module` and `load_module` are now deprecated

What changed in Python 3.4

Both `find_module` and `load_module` are now deprecated

Successors are `find_spec` and `create_module`

WHAT CHANGED IN PYTHON 3.4

Both `find_module` and `load_module` are now deprecated

Successors are `find_spec` and `create_module`

But

WHAT CHANGED IN PYTHON 3.4

Both `find_module` and `load_module` are now deprecated

Successors are `find_spec` and `create_module`

But

Both `find_module` and `load_module` work fine on both 2.7 and 3

VERY SIMPLE IMPORT BLOCKER

A basic example, blocking importing certain modules:

VERY SIMPLE IMPORT BLOCKER

A basic example, blocking importing certain modules:

```
import sys

class Blocker(object):
    def __init__(self, *args):
        self.modules = args

    def find_module(self, name, path=None):
        if name in self.modules:
            return self
        return None

    def load_module(self, name):
        raise ImportError("{} is blocked, yo".format(name))

sys.meta_path = [Blocker('ConfigParser')]
```

Here's one I made earlier

SUMMARY

There’s a lot you can do with import hooks

- Remote python modules

SUMMARY

There’s a lot you can do with import hooks

- Remote python modules
- encrypted python modules

SUMMARY

We talked a bit about import hooks. They're cool, yo

You should be able to get the slides from:

`kragniz.eu/pyconuk-2015.pdf`

Code for the json example I showed in the live demo is on github here:

`https://github.com/kragniz/json-sempai`

Follow me on github/whatever if you like this sort of thing!

LICENSE

Questions?