NAMESPACE PACKAGES

What are Implicit Namespace Packages?

Namespace packages are package-like

- directories
  - may contain modules
  - may contain nested regular packages
  - may contain nested namespace packages
  - but cannot contain `__init__.py`

These directories are implicitly made into these special types of packages

PEP 420

Mechanics

```
utils/
 validators/
   boolean.py
   date.py
   json/
     __init__.py
     serializers.py
     validators.py
```

`utils/` does not contains `__init__.py`	→ namespace package
`validators/` does not contain `__init__.py`	→ namespace package
`boolean.py` is a file with a `.py` extension	→ module
`json/` contains `__init__.py`	→ regular package
`serializers.py` is a file with a `.py` extension	→ module

Regular vs Namespace Packages

Regular Package

`type`	→ `module`
`__init__.py`	→ yes
`__file__`	→ package `__init__`
paths	→ breaks if parent directories change and absolute imports are used

single package lives in single directory

Namespace Package

`type`	→ `module`
`__init__.py`	→ no
`__file__`	→ not set
paths	→ dynamic path computation so OK if parent directories change

(your import statements will still need to be modified)

single package can live in multiple (non-nested) directories

in fact, parts of the namespace may even be in a zip file

Example

```
app/
  utils/
    validators/
      boolean.py
  common/
    __init__.py
    validators/
      boolean.py
```

	namespace package utils	*regular package* common
type	module	module
`__name__`	utils	common
`__repr__()`	`<module utils (namespace)>`	`<module common from '…/app/common'>`
`__path__`	`_Namespace(['…/app/utils'])`	`['…/app/utils']`
`__file__`	not set	`…/app/common/__init__.py`
`__package__`	utils	common
→ validators	utils.validators	common.validators

Import Examples

```
utils/
  validators/
     boolean.py
     date.py
     json/
        __init__.py
        serializers.py
        validators.py
```

```
import utils.validators.boolean

from utils.validators import date

import utils.validators.json.serializers
```

First familiarize yourself with regular packages.

Once you are <u>completely</u> comfortable with them, check out namespace packages if you want

Read PEP 420 – that should definitely be your starting point