nestly Documentation

Release 0.2

Erick Matsen et al.

March 25, 2014

Nestly is a small package designed to ease running software with combinatorial choices of parameters. It can easily do so for "cartesian products" of parameter choices, but can do much more– arbitrary "backwards-looking" dependencies can be used.

To find out more, look in the `examples/` subdirectory.

Contents:

CHAPTER 1

nestly Package

1.1 `nestly` Package

1.2 `core` Module

1.3 Subpackages

CHAPTER 2

Project Modules

2.1 nestly Package

2.1.1 `nestly` Package

2.1.2 `core` Module

2.1.3 Subpackages

CHAPTER 3

Examples

3.1 Building Nests

3.1.1 Basic Nest

From `examples/basic_nest/make_nest.py`, this is a simple, combinatorial example.

```
#!/usr/bin/env python

import glob
import math
import os
import os.path
from nestly import Nest

wd = os.getcwd()
input_dir = os.path.join(wd, 'inputs')

nest = Nest()

# Simplest case: Levels are added with a name and an iterable
nest.add('strategy', ('exhaustive', 'approximate'))

# Items can update the control dictionary
nest.add('run_count', [{'run_count': 10**i, 'function': 'pow'}
                       for i in xrange(3)], update=True)

# label_func is applied to each item create a directory name
nest.add('input_file', glob.glob(os.path.join(input_dir, 'file*')),
        label_func=os.path.basename)

# Items can be added that don't generate directories
nest.add('base_dir', [os.getcwd()], create_dir=False)

# Any function taking one argument (control dictionary) and returning an
# iterable may also be used:
def log_run_count(c):
    run_count = c['run_count']
    return [math.log(run_count, 10)]
nest.add('run_count_log', log_run_count, create_dir=False)

nest.build('runs')
```

3.1.2 Meal

This is quite a bit more complicated, with lookups on previous values of the control dictionary:

```
#!/usr/bin/env python

import glob
import os
import os.path

from nestly import Nest

wd = os.getcwd()
startersdir = os.path.join(wd, "starters")
winedir = os.path.join(wd, "wine")
mainsdir = os.path.join(wd, "mains")

nest = Nest()

bn = os.path.basename
def strip_bn(path):
    """Return the basename of path, any extension stripped"""
    return bn(os.path.splitext(path)[0])

# start by mirroring the two directory levels in startersdir, and name those
# directories "ethnicity" and "dietary"
nest.add('ethnicity', glob.glob(os.path.join(startersdir, '*')),
    label_func=bn)
nest.add('dietary', lambda c: glob.glob(os.path.join(c['ethnicity'], '*')),
    label_func=bn)

## now get all of the starters
nest.add('starter', lambda c: glob.glob(os.path.join(c['dietary'], '*')),
    label_func=strip_bn)
## now get the corresponding mains
nest.add('main', lambda c: [os.path.join(mainsdir, bn(c['ethnicity']) + "_stirfry.txt")],
    label_func=strip_bn)

## get only the tasty wines
nest.add('wine', glob.glob(os.path.join(winedir, '*.tasty')),
    label_func=strip_bn)
## the wineglasses should be chosen by the wine choice, but we don't want to
## make a directory for those.
nest.add('wineglass', lambda c: [strip_bn(c['wine']) + ' wine glasses'],
        create_dir=False)

nest.build('runs')
```

CHAPTER 4

Indices and tables

- *genindex*
- *modindex*
- *search*