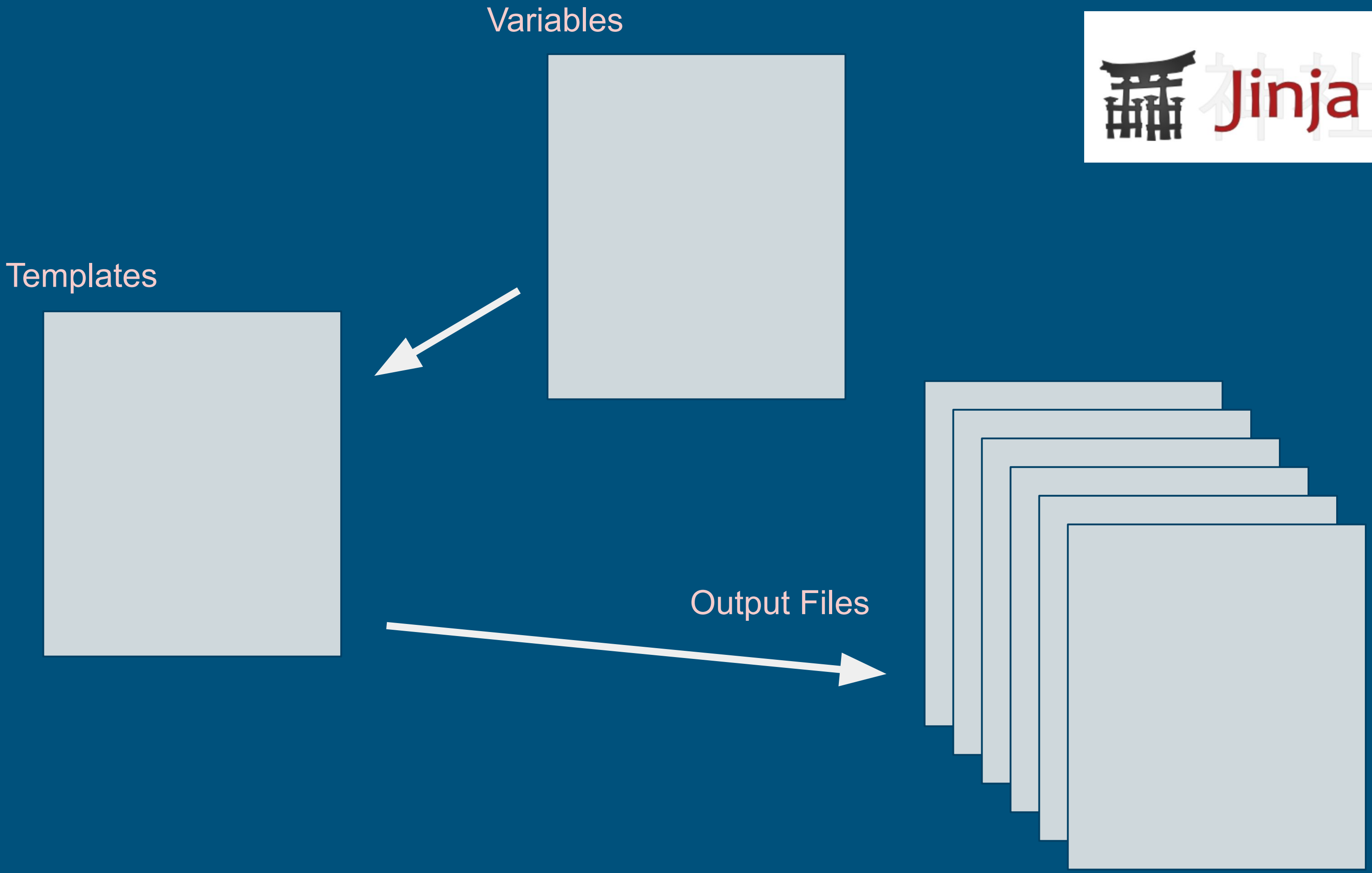
PYTHON
FOR NETWORK ENGINEERS

Onsite Training Session
June 2019

Day4 Schedule

- Review
- Jinja2 Templating
- Pulling data from a CSV file
- Integration to an Excel file
- Integrating to a Database
- Concurrency: Threads and Processes
- Unit Testing with pytest
- An introduction to Continuous Integration (optional)


Variables
Templates
Output Files

Jinja2 Templating

```
import jinja2

my_dict = {'a': 'whatever'}

my_template = '''
Some text
of something
{{ a }}
something
'''

t = jinja2.Template(my_template)
print(t.render(my_dict))
```

Reference Material in:

{{ github_repo }}/jinja2_example/jinja2_simple.py
{{ github_repo }}/jinja2_example/jinja2_bgp.py

```
Some text
of something
whatever
something
```

Jinja2 Templating
Loading Template from a File

```
import jinja2

template_file = 'bgp_config.j2'
with open(template_file) as f:
    bgp_template = f.read()

my_vars = {
    'peer_as': '22',
    'neighbor1': '10.10.10.2',
    'neighbor2': '10.10.10.99',
    'neighbor3': '10.10.10.220',
}

template = jinja2.Template(bgp_template)
print(template.render(my_vars))
```

Reference Material in:

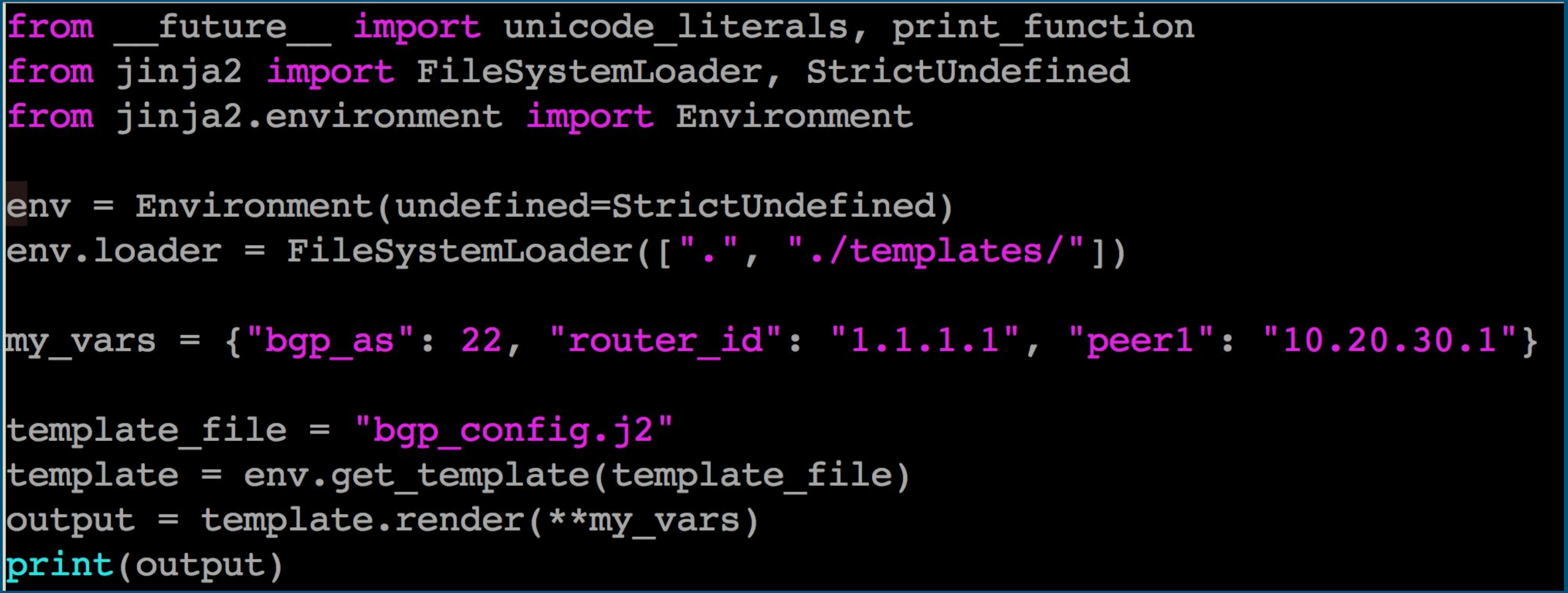
{{ github_repo }}/jinja2_example/jinja2_bgp_file.py

Exercises:
./day4/jinja2_ex1.txt

Reference Material in:
{{ github_repo }}/jinja2_example/jinja2_env.py

Jinja2 Template - Environment

Exercises:
./day4/jinja2_ex2.txt



```
from __future__ import unicode_literals, print_function
from jinja2 import FileSystemLoader, StrictUndefined
from jinja2.environment import Environment

env = Environment(undefined=StrictUndefined)
env.loader = FileSystemLoader([".", "./templates/"])

my_vars = {"bgp_as": 22, "router_id": "1.1.1.1", "peer1": "10.20.30.1"}

template_file = "bgp_config.j2"
template = env.get_template(template_file)
output = template.render(**my_vars)
print(output)
```

Jinja2 Templating - Conditionals

Exercises:
./day4/jinja2_ex3.txt

```
{% if SNMPv3 %}
access-list 98 remark *** SNMP ***
access-list 98 permit any
!
snmp-server view VIEWSTD iso included
snmp-server group READONLY v3 priv read VIEWSTD access 98
snmp-server user pysnmp READONLY v3 auth sha auth_key priv aes 128
encrypt_key
{% endif %}
```

Jinja2 Templating - Loops

```
protocols {
    bgp {
        group external-peers {
            type external;
            {% for neighbor_ip, neighbor_as in my_list %}
                neighbor {{ neighbor_ip }} {
                    peer-as {{ neighbor_as }};
                }
            {% endfor %}
        }
    }
}
```

Reference Material in:
{{ github_repo }}/jinja2_example/jinja2_bgp_loop.py

Jinja2 - Other Topics

- Jinja2 Whitespace Stripping
- Jinja2 Create Variables
- Jinja2 Filters
- Jinja2 Macros
- Jinja2 Includes / Hierarchy

CSV Examples

```
device_name,device_type,host,username,password
pynet-rtr1,cisco_ios,184.105.247.70,pyclass,my_pass
pynet-rtr2,cisco_ios,184.105.247.71,pyclass,my_pass
------------------------------------
file_name = 'test_net_devices.csv'
with open(file_name) as f:
    read_csv = csv.DictReader(f)
    for entry in read_csv:
        print(entry)
```

Reference Material in:
{{ github_repo }}/csv_example

Exercises:
./day4/csv_ex1.txt

Excel Examples

Reference Material in:
{{ github_repo }}/excel_example

Exercises:
./day4/excel_ex1.txt
./day4/excel_ex2.txt

```
from openpyxl import load_workbook


wb = load_workbook("excel_wb.xlsx")
print(f"Workbook Sheets: {wb.sheetnames}")
users_sheet = wb["Users"]
users_sheet.cell(row=5, column=3).value
```

Integrating to a DB

- Django ORM
- Defining the DB
- Creating the DB
- Primary Keys, Foreign Keys
- CRUD Operations

Reference notes in:
{{ github_repo }}/django/django_notes.txt

Defining the Database Fields (models.py)

```
class NetworkDevice(models.Model):
    device_name     = models.CharField(primary_key=True, max_length=80)
    device_type     = models.CharField(max_length=50)
    ip_address      = models.GenericIPAddressField()
    port            = models.IntegerField()
    vendor          = models.CharField(max_length=50, blank=True, null=True)
    model           = models.CharField(max_length=50, blank=True, null=True)
    os_version      = models.CharField(max_length=100, blank=True, null=True)
    serial_number   = models.CharField(max_length=50, blank=True, null=True)
    uptime_seconds  = models.IntegerField(blank=True, null=True)
    credentials     = models.ForeignKey(Credentials, blank=True, null=True)
```

Initializing the DB

```
cd ~/DJANGOX/djproject

$ python manage.py makemigrations
Migrations for 'net_system':
  0001_initial.py:
    - Create model Credentials
    - Create model NetworkDevice

$ python manage.py migrate
...
```

Exercises:
Initialize your Django Database
./day4/db_ex1a.txt

See:
./day4/db_ex1a_solution.txt

Create/Delete Objects

```
cd ~/DJANGOX/djproject/
$ python manage.py shell
...
>>> from net_system.models import NetworkDevice
>>> pynet_sw2 = NetworkDevice(
...         device_name='pynet-sw2',
...         device_type='arista_eos',
...         ip_address='184.105.247.73',
...         port=22,
... )
>>> pynet_sw2.save()
>>> pynet_sw2.delete()
>>> pynet_sw2 = NetworkDevice.objects.get_or_create(...)
```

PYTHON FOR NETWORK ENGINEERS

Load Data into the DB

```
$ cd ~/DJANGOX/djproject/net_system

$ python load_devices.py
(<NetworkDevice: NetworkDevice object>, True)
(<NetworkDevice: NetworkDevice object>, True)
(<NetworkDevice: NetworkDevice object>, True)
(<NetworkDevice: NetworkDevice object>, True)
(<NetworkDevice: NetworkDevice object>, True)
(<NetworkDevice: NetworkDevice object>, True)

$ python load_credentials.py
(<Credentials: Credentials object>, True)
(<Credentials: Credentials object>, True)
```

Exercises:
Load your data.
./day4/db_ex1b.txt

See:
./day4/db_ex1b_solution.txt

Query the Database

Exercises:
Manually query the database

See:
./day4/db_ex1c.txt

```
$ python manage.py shell
...
>>> from net_system.models import NetworkDevice
>>> all_devices = NetworkDevice.objects.all()
>>> all_devices
[<NetworkDevice: pynet-rtr1>, <NetworkDevice: pynet-rtr2>, <NetworkDevice: pynet-sw1>,
<NetworkDevice: pynet-sw2>, <NetworkDevice: pynet-sw3>, <NetworkDevice: pynet-sw4>,
<NetworkDevice: juniper-srx>]

>>> all_devices[0]
<NetworkDevice: pynet-rtr1>
>>> all_devices[0].ip_address
'184.105.247.70'
```

Link to credentials

```
>>> NetworkDevice.objects.get(ip_address='184.105.247.72')
<NetworkDevice: pynet-sw1>
>>> arista1 = NetworkDevice.objects.get(ip_address='184.105.247.72')

>>> from net_system.models import Credentials
>>> creds = Credentials.objects.all()
>>> creds
[<Credentials: pyclass>, <Credentials: admin1>]

>>> arista_creds = creds[1]
>>> arista1.credentials = arista_creds
>>> arista1.save()
```

Exercises:
./day4/db_ex1d.txt

Solution:
./day4/db_ex1d_solution.txt
./day4/db_ex1d.py

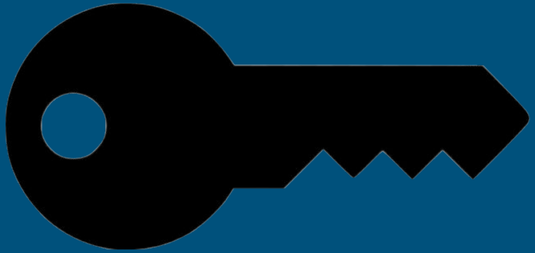
Retrieving all objects using a given credential

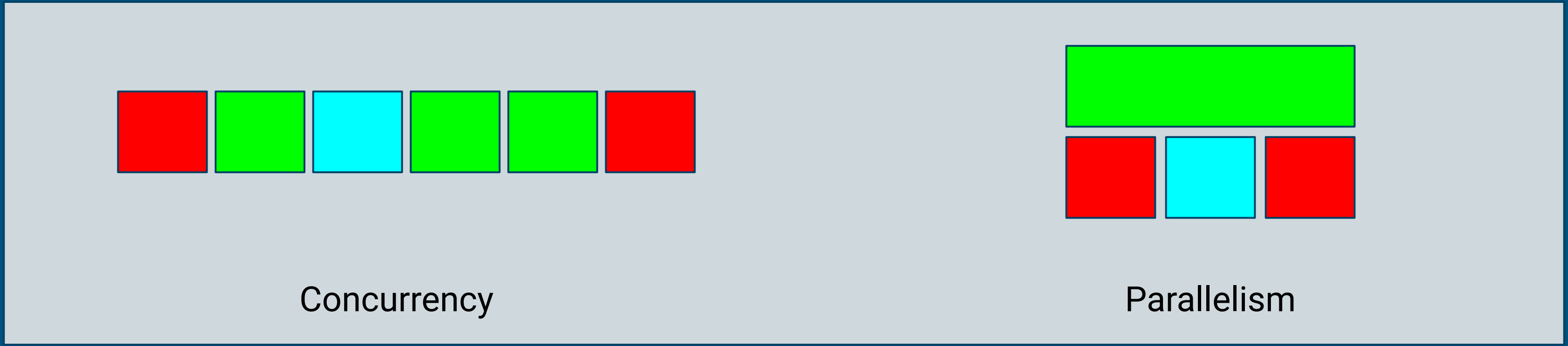
```
>>> arista_creds
<Credentials: admin1>

>>> arista_creds.networkdevice_set.all()
[<NetworkDevice: pynet-sw1>, <NetworkDevice: pynet-sw2>]
```

Exercises:
./day4/db_ex2.txt
./day4/db_ex3.txt
./day4/db_ex4.txt

Concurrency/Parallelism

- Concurrency? Parallelism?
- Python and the GIL
- Concurrent Futures





Concurrent Futures

- Python 3.2 + backported to Python 2
- Wrapper around Threading/Processes
- Provides consistent interface using either Threads or Processes -- meaning very easy to switch concurrency method
- Threads: for I/O bound things (waiting for stuff in the network)
- Processes: for CPU bound things (crunch lots and lots of numbers)

Concurrent Futures - ThreadPool

Reference Material in:
{{ github_repo }}/concurrency_example

Exercises:
./day4/concurrency_ex1.txt

```
from concurrent.futures import ThreadPoolExecutor


pool = ThreadPoolExecutor(max_workers=8)
futures_threads = []
for _ in range(10):
    futures_threads.append(pool.submit(some_func))
```

Concurrent Futures - ProcessPool

Reference Material in:
{{ github_repo }}/concurrency_example

Exercises:
./day4/concurrency_ex2.txt

```
from concurrent.futures import ProcessPoolExecutor


pool = ProcessPoolExecutor(max_workers=8)
futures_procs = []
for _ in range(10):
    futures_procs.append(pool.submit(some_func))
```

Concurrent Futures - As Completed & Wait

Reference Material in:
{{ github_repo }}/concurrency_example

Exercises:
./day4/concurrency_ex3.txt

```
from concurrent.futures import ProcessPoolExecutor, as_completed, wait


pool = ProcessPoolExecutor(max_workers=8)
futures_procs = []
for _ in range(10):
    futures_procs.append(pool.submit(some_func))
for proc in as_completed(futures_procs):
    print(proc.result())
wait(futures_procs)
```

Writing Reusable Code/Thinking in terms of a System

- Functions/Classes
- Code Structure
- Linting Tools
- Unit Testing
- Systems Testing
- CI-CD

Unit Testing

```
import pytest

# Functions
def func(x):
    return x + 1

# Tests
def test_answer():
    assert func(3) == 4
```

PYTHON
FOR NETWORK ENGINEERS

Reference Material in:
{{ github_repo }}/unittest_example

Unit Testing

```
$ py.test -s -v ./test_simple.py

==================== test session starts ====================================
platform linux -- Python 3.6.8, pytest-4.6.2, py-1.8.0, pluggy-0.12.0 --
/home/student35/VENV/py3_venv/bin/python36
cachedir: .pytest_cache
rootdir: /home/student35/pynet-ons-ds/testing_example/pytest_dir
plugins: pylama-7.7.1, f5-sdk-3.0.21
collected 2 items

test_simple.py::test_answer PASSED
test_simple.py::test_answer2 PASSED

================= 2 passed in 0.01 seconds ====================================
```

But my unit tests work…

Creating a fixture

```
@pytest.fixture(scope="module")
def netmiko_connect():
    cisco1 = {
        'device_type': 'cisco_ios',
        'ip':   '184.105.247.70',
        'username': 'pyclass',
        'password': getpass()
    }
    return ConnectHandler(**cisco1)
```

Using a fixture

Exercises:
./day4/unittest_ex1.txt
./day4/unittest_ex2.txt

```
def test_prompt(netmiko_connect):
    print(netmiko_connect.find_prompt())
    assert netmiko_connect.find_prompt() == 'pynet-rtr1#'

def test_show_version(netmiko_connect):
    output = netmiko_connect.send_command("show version")
    assert 'Configuration register is 0x2102' in output
```

If it doesn't happen automatically; it didn't happen.

GitLab

Travis CI

Azure Pipelines

circleci

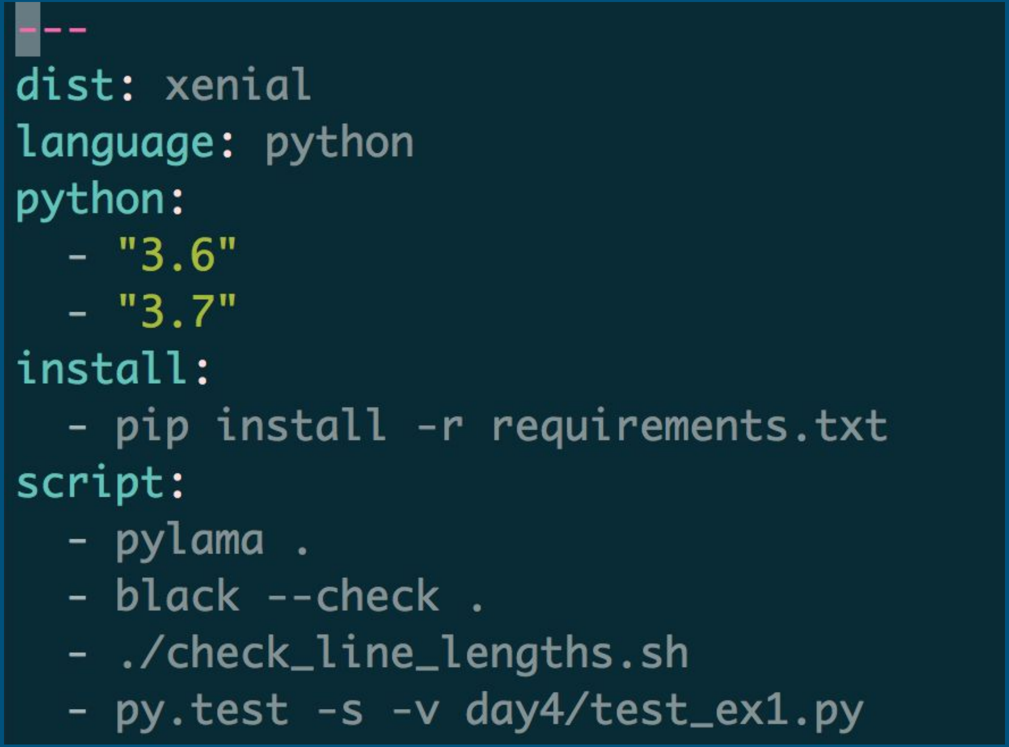
Continuous Integration using Travis CI

Define a .travis.yml file in your repository.

Link Travis-CI to GitHub account

Add linting

Add automated testing



```yaml
---
dist: xenial
language: python
python:
  - "3.6"
  - "3.7"
install:
  - pip install -r requirements.txt
script:
  - pylama .
  - black --check .
  - ./check_line_lengths.sh
  - py.test -s -v day4/test_ex1.py
```

The end...

Questions?

ktbyers@twb-tech.com
Twitter: @kirkbyers