Code Analysis and Testing Training

Python use case

Olivier Sallou - CC-BY-SA

Why?

Testing = reproductibility

Code analysis:

- follow best practices
- detect some errors before they occur (mainly dynamic languages)

Tests

Unit testing: test methods (isolated)

Functional testing: test behavior of software (alone)

Integration testing: test software in “real” environment (with other software/servers)

Behavior driven development/testing: describe test as “features”

Clean environment

Test in a clean fresh new environment

- virtualenv (does not manage system deps)
- continuous integration servers

Virtualenv

```
# virtualenv mysoftware
# . mysoftware/bin/activate
# pip install xx
```

Compile/Install scripts

setup.py : describes software and dependencies

easy/reproductible build/install

dependencies

upload to pypi

Unit test

nose: executes a list of test (unit / functional tests)

Provides pre/post test methods for easier setup (create user, clean db, …)

Nose

#pip install nose

#nosetest

Mocks

Simulate an object/class that cannot be tested in the environment (a LDAP for example).

Behavior driven tests

```
Feature: showing off behave

  Scenario: run a simple test
     Given we have behave installed
      when we implement a test
      then behave will test it for us!
```

For each features, describe scenarii like human readable sentences

They should match your backlog.

```
from behave import *

@given('we have behave installed')
def step_impl(context):
    pass

@when('we implement a test')
def step_impl(context):
    assert True is not False

@then('behave will test it for us!')
def step_impl(context):
    assert context.failed is False
```

Continous Integration

git post-commit

github

=> travis, drone.io, ...

Code coverage

Check the code covered by your tests

=> do not target 100%

=> try to match most of the code and important parts

nosetests --with-coverage --cover-package=mysamplecode

Static code analysis

Checks code according to best practices:

- unused variables, spaces, line length, ...

prospector => prospector --no-autodetect

pylint => pylint mysamplecode

pep8 => pep8 mysamplecode

sonar (needs a sonar server)

Documentation

Generate documentation for libraries

sphinx