ORACLE

Testing Cloud Native Applications

Prasenjit Sarkar
Oracle Cloud Infrastructure
February 2020

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Objectives

After completing this lesson, you should be able to;

- Describe different Testing Methodologies
- Describe various ways to test Cloud Native Applications

ORACLE

Testing Cloud-Native Applications

Testing

- Can't survive with manual tests
- Reliable, repeatable and automated tests are essential
- Use test doubles (mocks, fakes and stubs) for dependencies

MOCK

Object use to test the calls it receives

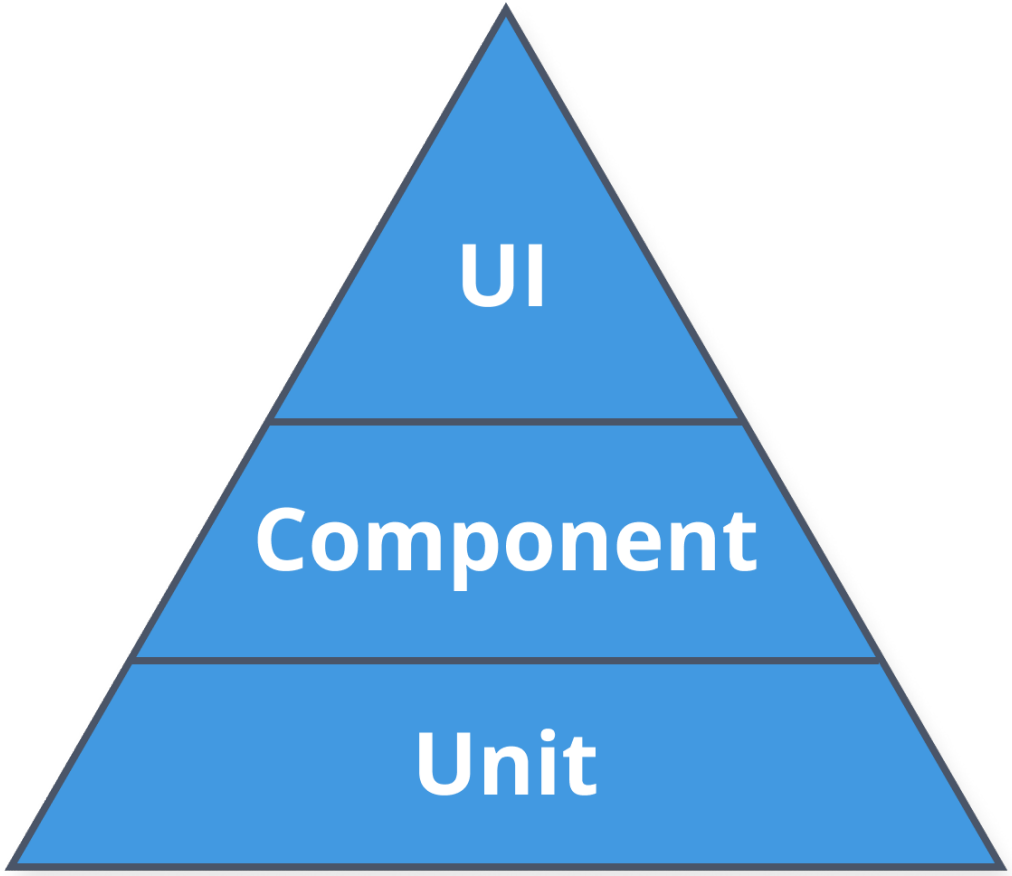
FAKE

Object with working implementation (di"erent from a real implementation)

STUB

Object that returns prepopulated data

Test Automation Pyramid



Building blocks of Testing

Unit tests 

- Fast to execute First line of defense

Service Tests / Component-level tests 

- Test interactions between services/components

User interface tests 

- End-to-end tests
- (Relatively) slow to execute and costly to write/maintain Crucial for usability/accessibility

Different type of Testing

- Acceptance tests
- Smoke tests
- Integration tests
- Security
  - Penetration tests
  - Fuzz tests
- Performance tests
  - Load tests
- Usability tests
- Chaos tests
- Canary tests
- A/B tests

Right time of Testing

When and how often to execute tests?

Unit: during development, before every merge/check-in
Service/Component: before/after every merge/check-in
Integration: before deployments
Canary: run continuously *
Security: automated, part of integration/canary tests if possible

---

UI: on UI changes, consider automating if UI heavy
Performance: get baselines manually, consider automating for repeatable numbers A/B: as needed, make sure you have clear goals and metrics defined
Chaos: as needed, for operational readiness, to catch potential prod issues

* **can be costly to maintain**

Testing Environments

Dev → Test → Staging → Production

- Mimic production environment (as closely as possible)
- How to keep environments in sync?
- What's the cost for all this?
- Are you running staging in all regions, just like production

Additional Information

- For additional information on Testing Cloud Native Applications, please refer to:

https://www.oreilly.com/library/view/cloud-native-infrastructure/9781491984291/ch06.html

ORACLE

Oracle Cloud always free tier:
oracle.com/cloud/free/

OCI training and certification:
cloud.oracle.com/en_US/iaas/training
cloud.oracle.com/en_US/iaas/training/certification
education.oracle.com/oracle-certification-path/pFamily_647

OCI hands-on labs:
ocitraining.qloudable.com/provider/oracle

Oracle learning library videos on YouTube:
youtube.com/user/OracleLearning

Thank you