Improving Developer Productivity through

Continuous Regression Testing

Denver Java User Group, April 13, 2022

Pejman Ghorbanzade

Format

- Interactive
- Hands-on Live Coding
- Ask questions any time

Agenda

- Motivation
- Snapshot Testing
- Regression Testing
- Continuous Testing

About Me

- 6 Years of Experience
  - VMWare Carbon Black
  - Canon Medical Informatics
- Working full-time on touca.io
  - Continuous Regression Testing
- Passionate about maintaining software at scale

Image courtesy of Professor Prokop
RadboudUMC, Nijmegen, the Netherlands

Software Engineering

- Programming
  - Theoretical problem solving
  - Like sport
- Software Engineering
  - Problem solving within business constraints
  - Like gardening

> Software Engineering is programming integrated over time

The Building that Moved

Business Value

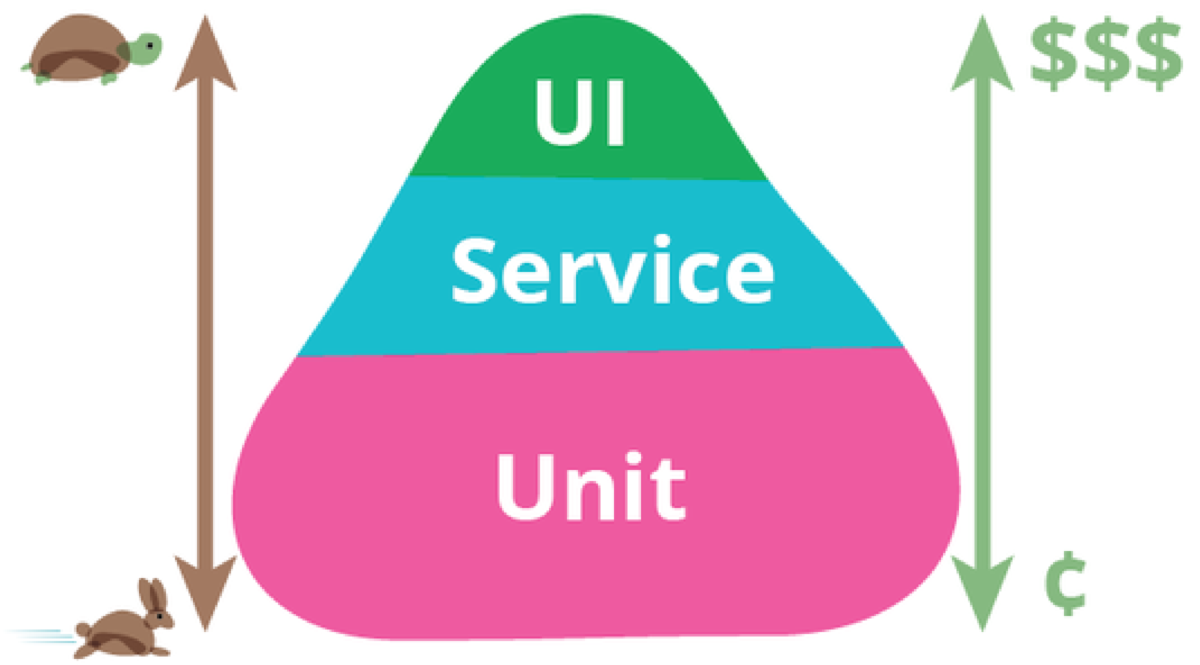
- Think like an engineer
  - Civil engineering: Building a house
  - Software engineering: Building with mud

> Software is a tractable medium.

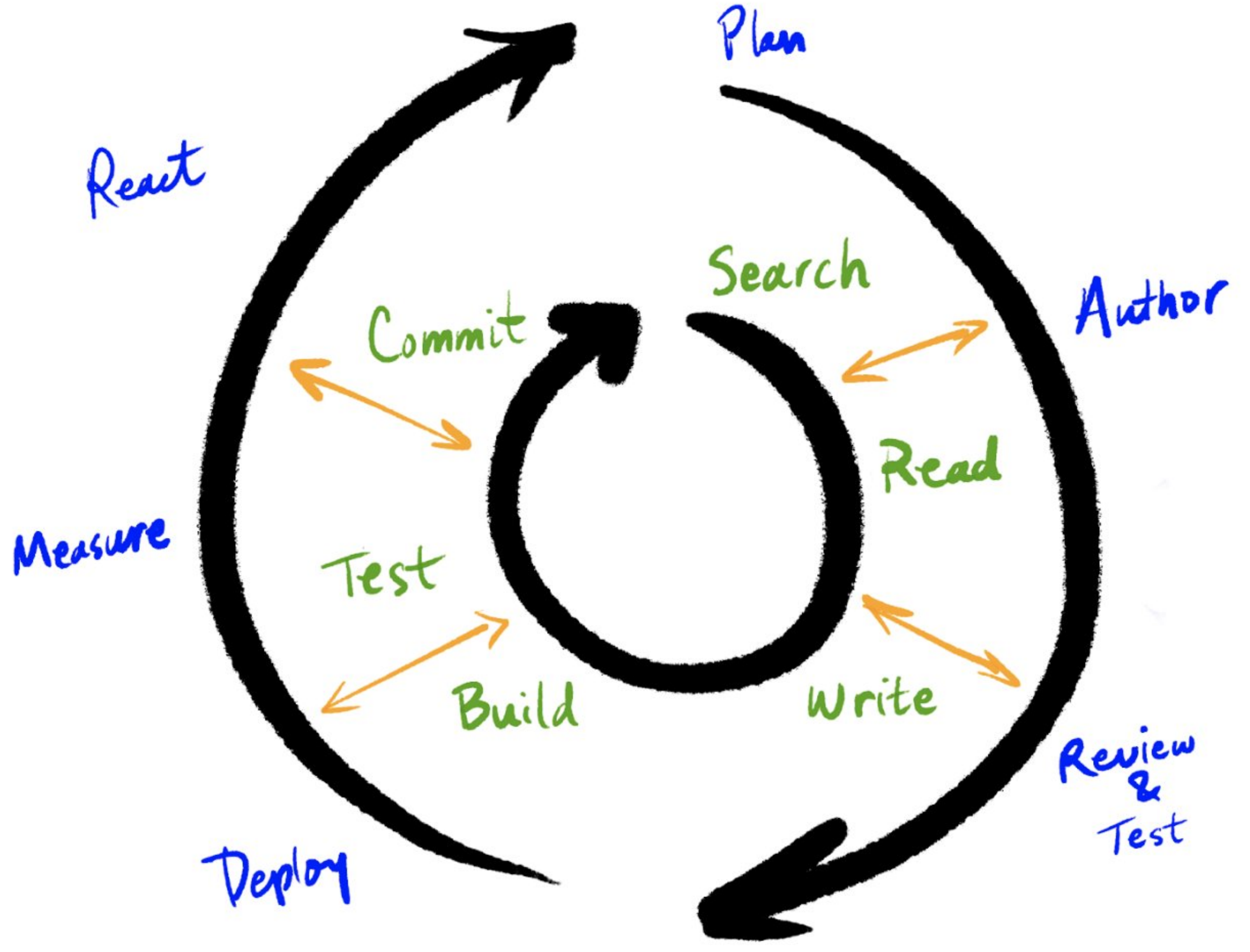
Software Testing Pyramid

- Good tests are:
  - Cheap to Write
  - Easy to Read
  - Fast to Run
  - Easy to Change

> Good tests have high return on investment.



Developer Inner Loop, Outer Loop



Finding bugs after deployment	💰💰💰💰💰
Finding bugs before release	💰💰💰
Finding bugs during QA testing	💰💰
Finding bugs during code review	💰
Finding bugs during development	

It takes 23 days for software engineers to gain confidence that a given code change works as they expect.

The Problem

How can we refactor half a million lines of code without causing any side effects?

Candidate Solution A

```
Output newOutput = newSystem(testcase);
Output oldOutput = oldSystem(testcase);
compare(newOutput, oldOutput);
```

Disadvantages

- Test is difficult to setup
- Test system is inefficient to run
- Test system is not reusable

Candidate Solution B

```
Output newOutput = newSystem(testcase);
File newFile = writeToFile(testcase, newOutput);
File oldFile = findOldFile(testcase);
compare(newFile, newOutput);
```

Disadvantages

- Dealing with files is no fun
- Test system is hard to maintain
- Test system is not reusable

Demo Time

Approval Testing

Candidate Solution C

```
final Output newOutput = newSystem(testcase);
final Description newDescription = describe(testcase, newOutput);
submit(testcase, newDescription);
```

Disadvantages

- Limited customization
- Overkill for small projects
- Requires remote computing resources

Simple Example

```java
public record Student(
  String username,
  String fullname,
  LocalDate dob,
  double gpa
) {}

public static Student findStudent(final String username) {
  // ...
}
```

Motivation

High-level API

```
import io.touca.Touca;

public final class StudentsTest {

  @Touca.Workflow
  public void findStudent(final String username) {
      Student student = Students.findStudent(username);
      Touca.assume("username", student.username);
      Touca.check("fullname", student.fullname);
      Touca.check("birth_date", student.dob);
      Touca.check("gpa", student.gpa);
  }

  public static void main(String[] args) {
    Touca.run(StudentsTest.class, args);
  }
}
```

Design Requirements

- Intuitive developer experience
- Intrinsic support for common types
    - Must support integral types, fractional types, Strings, Iterables, and other common standard types
- Extensible design to support user-defined types
    - Must allow users to introduce logic for handling custom types

Demo Time

Regression Testing

Questions

- https://touca.io
- https://github.com/trytouca/touca-java
- https://twitter.com/heypejman
- pejman@touca.io