Capstone

Automating BDD Scenarios

Admin

- Project:
  - If I have OKed your group's proposal, write me first scenarios and then BDD tests for your projects (one submission per student this time)
  - JBehave/cucumber-jvm for Java
  - Behave for python
- Again Credit:
  - Most images in this lecture from BDD in Action by John Smart

Benefits of Automated Scenarios

- Benefits of automated acceptance testing
  - Testers spend less time of regression testing
  - New versions released faster
  - Automated scenarios give accurate 'vision' of current state of project

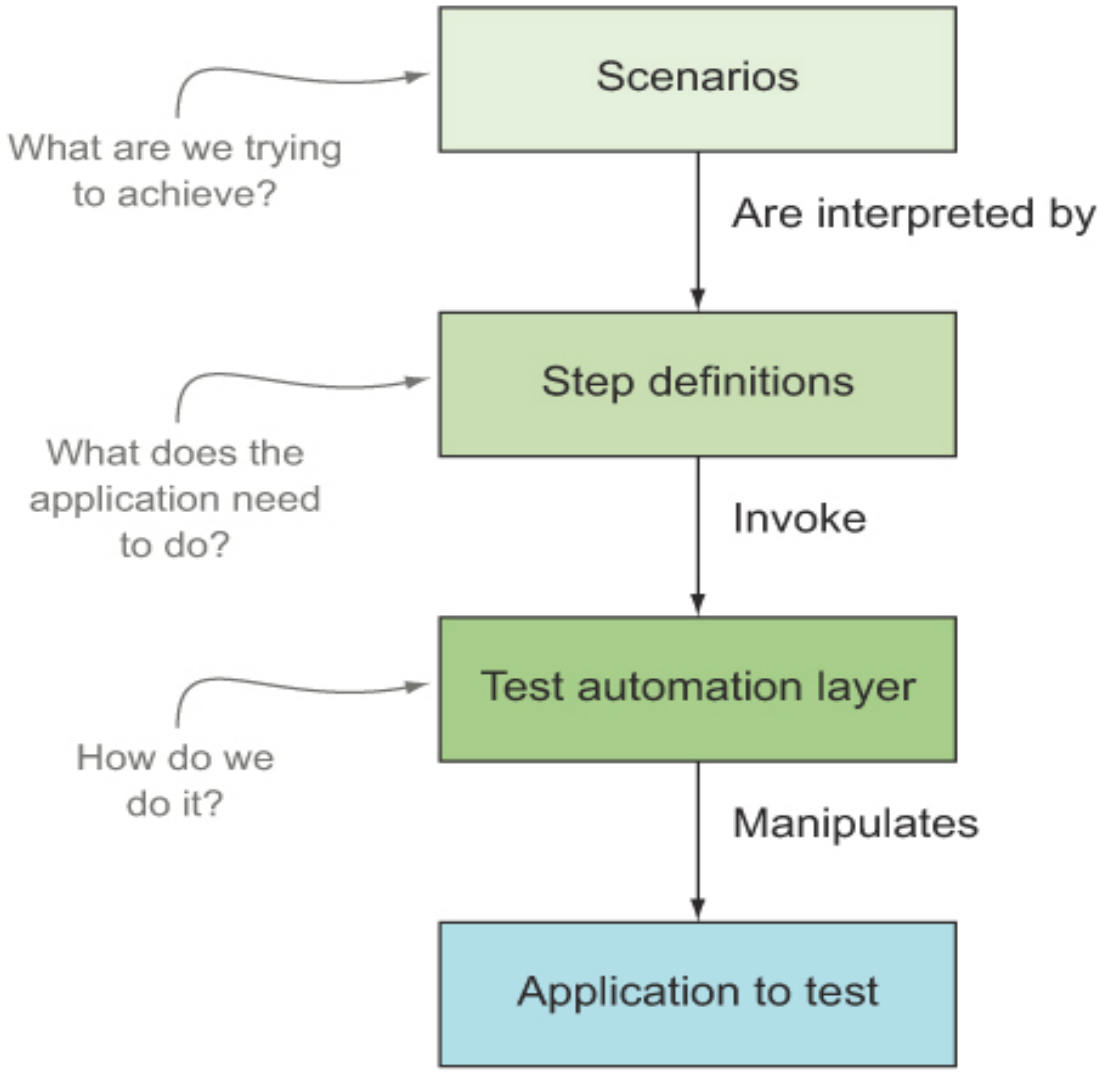
Continuous Integration and Testing

- https://www.youtube.com/watch?v=RcTFpNlkiUs
- For first section

Layers in Test Automation

- Got Scenarios
- Now step defs

Figure 6.3. A well-designed test automation suite has a number of layers.



Test Automation Layers Example

- This is how it might look
-

- *Scenarios* describe the high-level requirement:

```
Scenario: Earning standard points from an Economy flight
Given the flying distance between Sydney and Melbourne is
878 km
And I am a standard Frequent Flyer member
When I fly from Sydney to Melbourne
Then I should earn 439 points
```

- *Step definitions* interpret the scenario texts and call the test automation layer to perform the actual tasks:

```
@Given("the flying distance between $a and $b is $distance
km")
public void defineTheFlyingDistanceForATrip(...) {...}
```

- *The test automation layer* interacts with the application under test:

```
inTheTestDatabase.theDistanceBetween(departure)
                 .and(destination)
                 .is(distance);
```

Scenario Outcomes

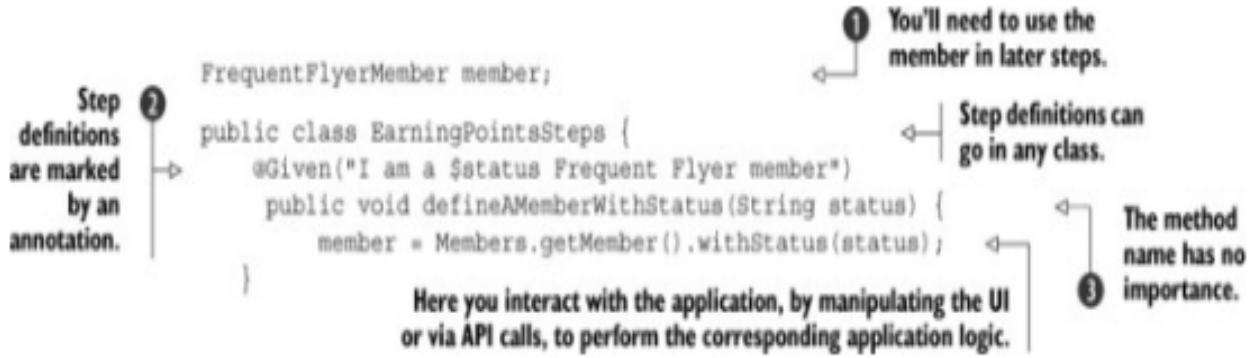
- Standard tests (TDD unit tests)
  - Two outcomes
  - Which are?

Scenario Outcomes

- Standard tests (TDD unit tests)
  - Two outcomes
    - Red bar (fail)
    - Green bar(pass)
- BDD Tests
  - Add more
    - Pending (throw new PendingException)
    - Skipped (tests after failed or pending test)
    - work-in-progress

JBehave Step Definition

- Example step definition
- Or from table




```
FrequentFlyerMember member;

public class EarningPointsSteps {
    @Given("I am a $status Frequent Flyer member")
     public void defineAMemberWithStatus(String status) {
         member = Members.getMember().withStatus(status);
     }
```


```
SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy");

@Given("I have travelled on the following flights: $flights")
public void travelled_on_flights(ExamplesTable flights)
  throws ParseException {
     for(Map<String,String> flightDetails : flights.getRows()) {
         Flight flight = Flight.number(flightDetails.get("flight"))
               .from(flightDetails.get("from"))
               .to(flightDetails.get("to"));

         Date date = formatter.parse(flightDetails.get("date"));
         member.flewOnFlight(flight).on(date);
     }
  }
```

In Python using behave

- Using behave
-

```
Feature: Transfer points to other members
  @transfers
  Scenario: Transfer points between existing members
    Given the following accounts:
      | owner | points | statusPoints |
      | Jill  | 100000 | 800          |
      | Joe   | 50000  | 50           |
    When Joe transfers 40000 points to Jill
    Then the accounts should be the following:
      | owner | points | statusPoints |
      | Jill  | 140000 | 800          |
      | John  | 10000  | 50           |
```

Behave will pass the tabular data to each step in the context object, as illustrated here:

```
@given('the following accounts')
def step_impl(context):
    for row in context.table:
        owner = row["owner"]
        points = row["points"]
        statusPoints = row["statusPoints"]
```

Tabular data is passed in the context.table variable.

Extract values from each row.

Side note: Tools

- In academia
  - Often testing/learning concepts
  - Often in isolation
  - Long debate:
    - What level of tools should we teach
    - When is it worth the time to invest in tools
- In industry:
  - Going to be working on a project for months not day/weeks
  - Tools always worth learning
  - Startup vs steady state

Tools often used in java BDD

- Tools for Java BDD
  - Often use maven
    https://maven.apache.org/download.cgi
    - Maven is make for java
    - Make?
  - JBehave package
    - Simple tutorial
    - http://jbehave.org/reference/stable/getting-started.html
  - Cucumber:
    - Good tutorial
    - http://www.hascode.com/2014/12/bdd-testing-with-cucumber-java-and-junit/

Python and BDD

- Behave library is the goto in python BDD
  - Nice tutorial
    - http://code.tutsplus.com/tutorials/behavior-driven-development-in-python--net-26547
  - The official tutorial
- But lettuce is getting popular
  - Java:cucumber as python:lettuce
  - http://lettuce.it/tutorial/simple.html
  -