

Using Acceptance Test-Driven Development / Behavior Driven Development in Context with Domain Driven Design and Clean Architecture

Ken Pugh

September 2022


1

Objectives

- Overview of Acceptance Test-Driven Development / Behavior Driven Development
- Learn how ATDD/BDD works with Domain Driven Design
- Learn how automating ATDD/BDD scenarios for testing works with Hexagonal / Clean Architecture

2

Ken Pugh



- ATDD/BDD, TDD, BVDD, Lean, Scrum, Kanban, Technical Excellence
- Over 2/5 century of software development experience
- Co-author SAFe® Agile Software Engineering
- Author of seven books, including:
 - *Lean Agile Acceptance Test-Driven Development: Better Software Through Collaboration*
 - *Prefactoring: Extreme Abstraction, Extreme Separation, Extreme Readability*
 - *Interface Oriented Design*

Helping teams deliver software more effectively

3

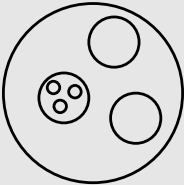
Overall Rule

There are exceptions to every statement, except this one

4

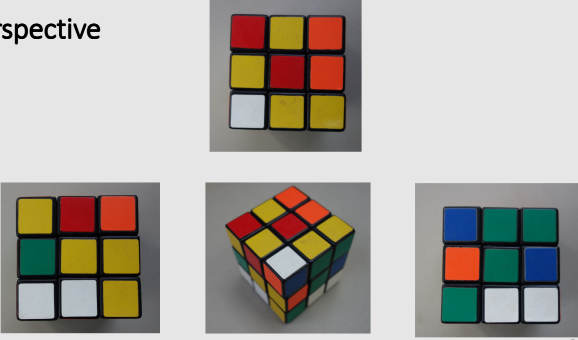
2nd Overall Rule

Context is everything
Everything exists in a context
Everything is always true in some context



5

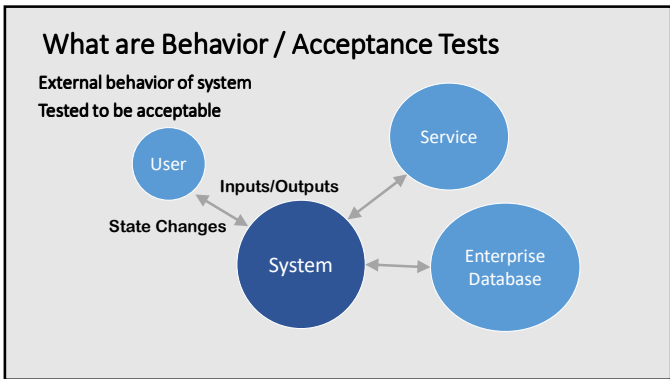
Perspective



6

Introduction

7

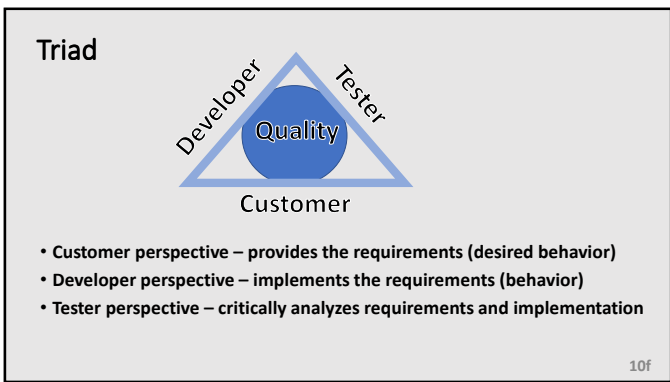


8

Definitions

- **Acceptance criteria**
 - General ideas
- **Acceptance tests**
 - Specific tests that either pass or fail
 - Implementation independent

9



10

ATDD/BDD

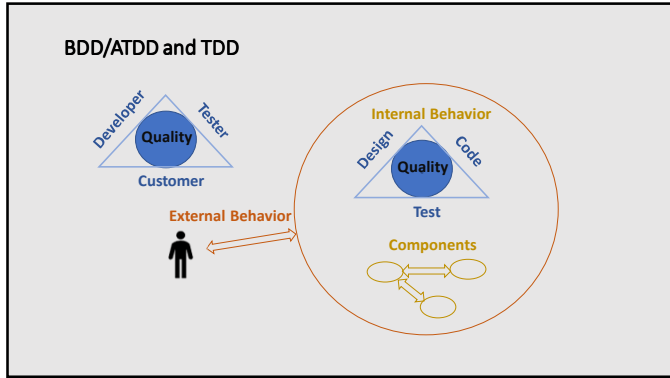
- **Behavior Driven Development (BDD)**
 - Define behavior of system which is tested
- **Acceptance Test Driven Development (ATDD)**
 - Create tests for acceptable behavior of the system

11

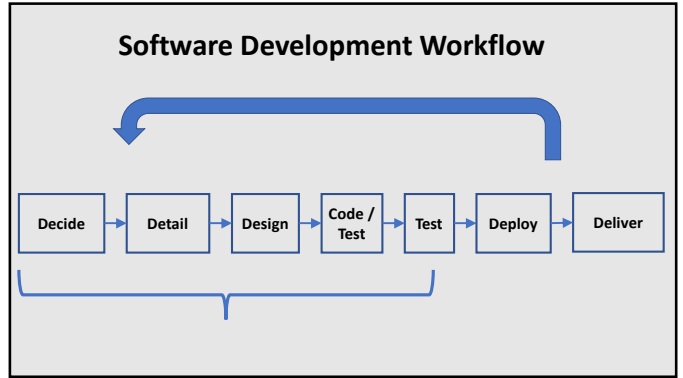
Goal of ATDD/BDD:

Replace misunderstanding
with
Shared understanding

12



13



14

Decide

15

Hypothesis Driven Development

- We think that an application that keeps track of manual test runs will decrease effort in running the tests by 5% as measured by the number of hours expended on manual testing.

16

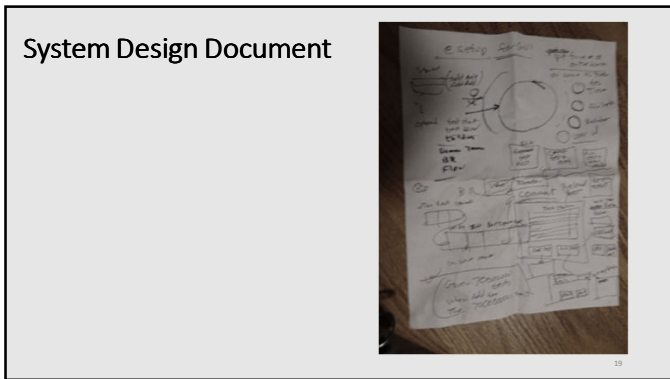
Feature / Story

- As a user, I want to be able to record test results for a manual test, so that I know which tests have been run at any time.

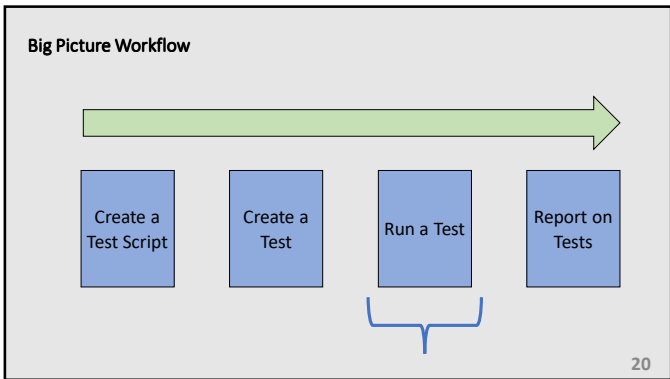
17

High Level Detail

18



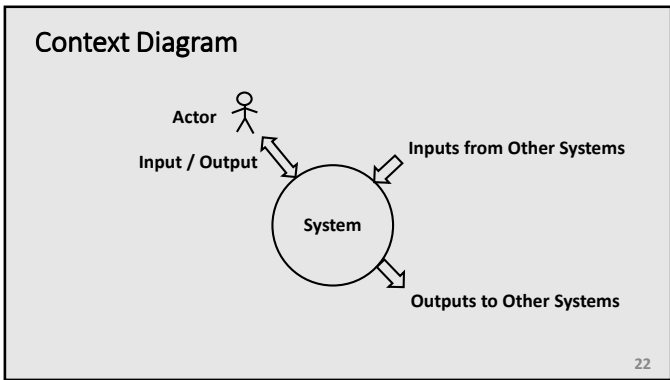
19



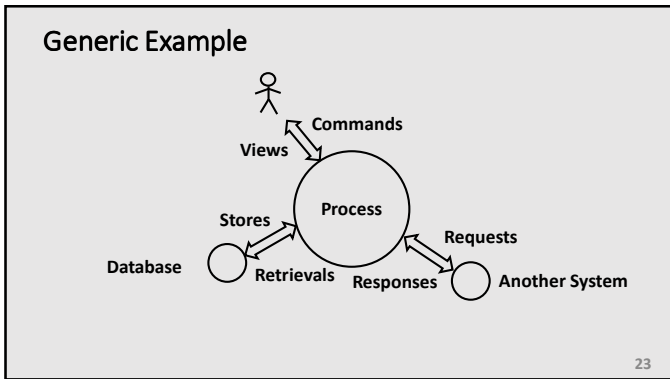
20

Context is Everything

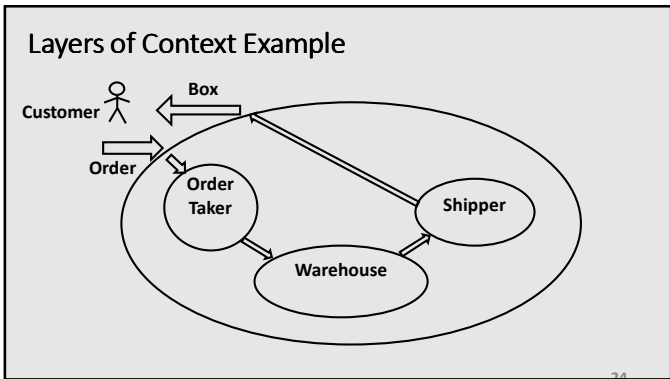
21



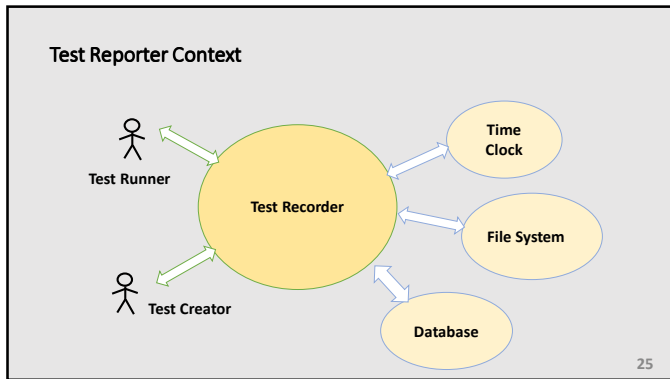
22



23

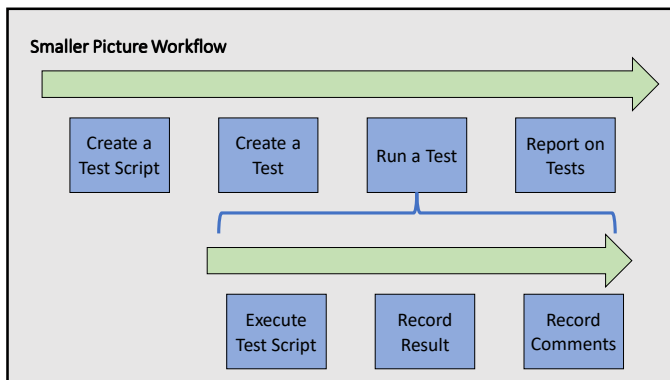


24



Story Details

26



Behavior

Types of behavior:

- Flow
- Domain Terms
- Business Rules

28

Scenario for Flow

Scenario is one path through a use case
(main or exception or alternative)

Scenario: <Name>
Given <current state> (preconditions)
When <event or action (ala Domain Event)occurs>
Then <new state, output, or both> (post-conditions)

29

Main Flow - Outline

Scenario: Run a test successfully
Given test exists
When test is run
Then test is updated with results of the run

30

Main Flow Version 1 (1)

Scenario: Run a test successfully

Given test exists

Issue ID	Name	Runner	Last Result
12345	Enter test result		Failure
...			
Date Last Run	Date Previous Result		
Never	Never		
...			
File Path	Comments		
EnterTest.feature			

31

31

Main Flow Version 1 (2)

And test is run

Result	Success	
Comments	Works great	
Runner	Sam	
Date Time	Oct 1, 2022, 12:30:01 AM	

32

32

Main Flow Version 1

Then test is now

Issue ID	Name	Runner	Last Result
12345	Enter test result	Sam	Success
...			
Date Last Run	Date Previous Result		
Oct 1, 2022, 12:30:01 AM	Never		
...			
File Path	Comments		
EnterTestResult.feature	Works great		

33

33

Scenarios Yield Information

- Domain Terms (ala Value Objects)
- Entities
 - Composite of attributes which are all domain terms
- Entity Collections
 - Typically have persistence (ala Repositories)

34

One Way To Implement

- Domain Terms declared as
 - Domain Term Types (abstract data types) having
 - toString()
 - From String : constructor(String value) / Class.parse(String value)
 - May have multiple Domain Terms of same Domain Term Type
- Entity
 - Attributes are Domain Terms
- EntityDTO
 - Attributes are all strings – initialized to appropriate values
 - To communicate with external world
- Entity Collection
 - Persistence Could use List<MyType> or Collection<MyType>

35

35

Domain Term

Scenario: Domain Term Result

* Result values are

Success	
Failure	

36

36

Domain Term Type

Scenario: Domain Term Type IssueID

Domain Term "Issue ID" is this type

* IssueID must be five characters and digits without spaces

Value	Valid	Notes
12345	Yes	
A1234	Yes	
1 123	No	Has spaces
1234	No	Too short
123456	No	Too long

37

Business Rules

Rule: Update Test from Test Run

Variation: Test Run Result Different From Previous

Variation: Test Run Result Same as Previous

Variation: Test Has Never Been Run

38

Business Rules (and Calculations)

Scenario: Update Test from Test Run

* Update Test from Test Run

Old Last Result	Old Date Last Run	Old Date Previous Result	New Last Result	New Date Last Run	New Date Previous Result
Success	Oct 1, 2022, 12:30:01 AM	Never	Success	Oct 1, 2022, 12:30:01 AM	Never
Failure	Oct 1, 2022, 12:30:01 AM	Never	Failure	Oct 1, 2022, 12:30:01 AM	Never
Success	Oct 1, 2022, 12:30:02 AM	Never	Success	Oct 1, 2022, 12:30:02 AM	Never
Failure	Oct 1, 2022, 12:30:02 AM	Never	Failure	Oct 1, 2022, 12:30:02 AM	Never
Success	Oct 1, 2022, 12:30:03 AM	Never	Failure	Oct 1, 2022, 12:30:03 AM	Never

39

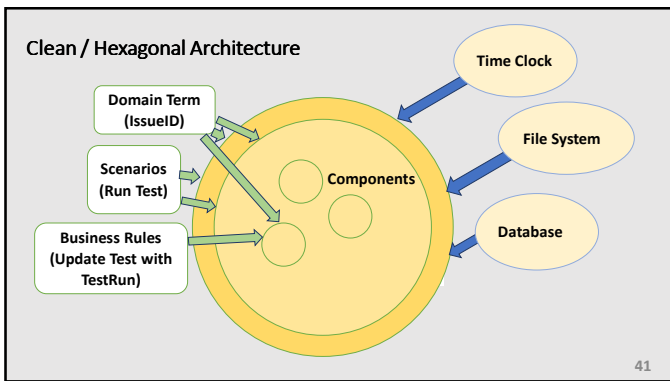
Business Rule Zoomed

Scenario: Update Test from Test Run

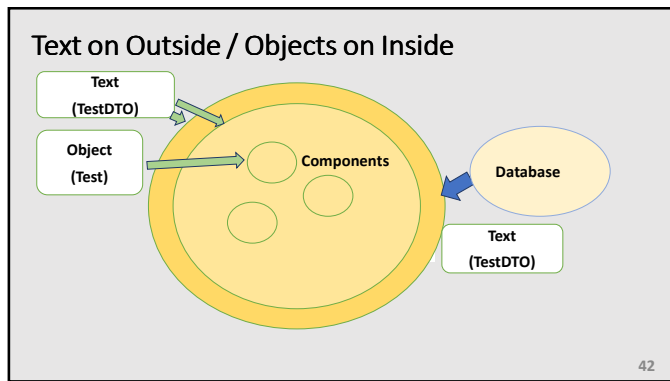
* Update Test from Test Run

Old Last Result	Old Date Last Run	Old Date Previous Result	New Last Result	New Date Last Run	New Date Previous Result
Success	Oct 1, 2022, 12:30:01 AM	Never	Success	Oct 1, 2022, 12:30:01 AM	Never
Failure	Oct 1, 2022, 12:30:01 AM	Never	Failure	Oct 1, 2022, 12:30:01 AM	Never
Success	Oct 1, 2022, 12:30:02 AM	Never	Success	Oct 1, 2022, 12:30:02 AM	Never
Failure	Oct 1, 2022, 12:30:02 AM	Never	Failure	Oct 1, 2022, 12:30:02 AM	Never
Success	Oct 1, 2022, 12:30:03 AM	Never	Failure	Oct 1, 2022, 12:30:03 AM	Never

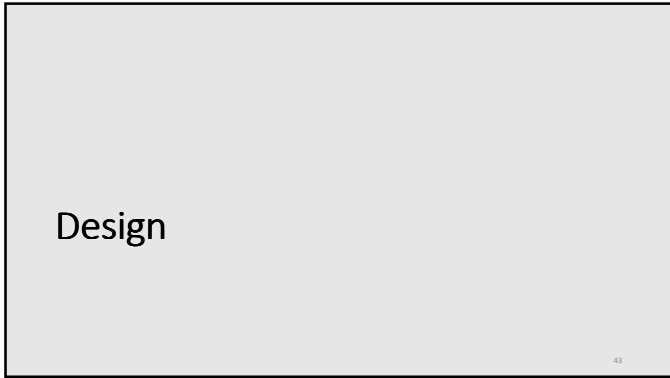
40



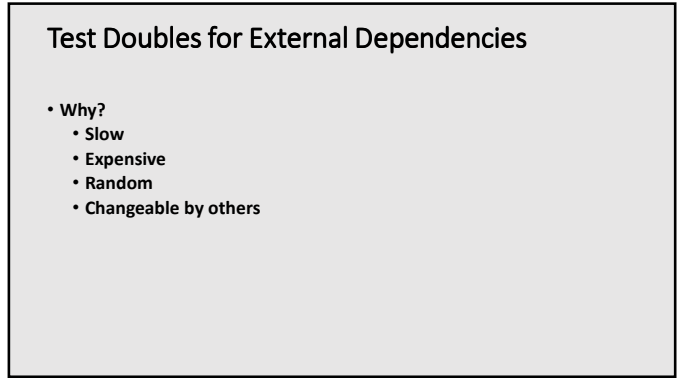
41



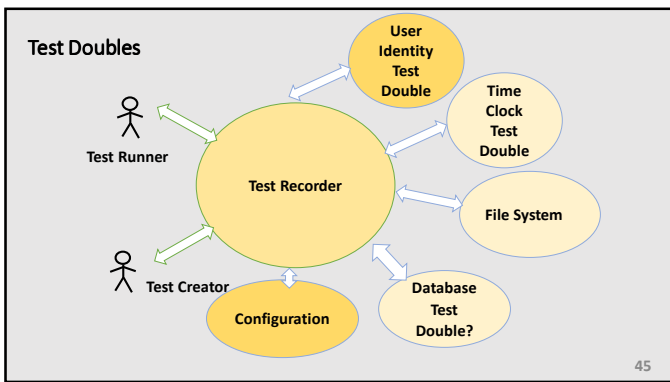
42



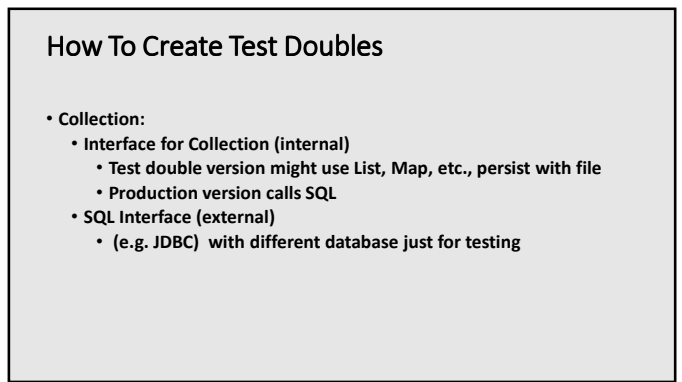
43



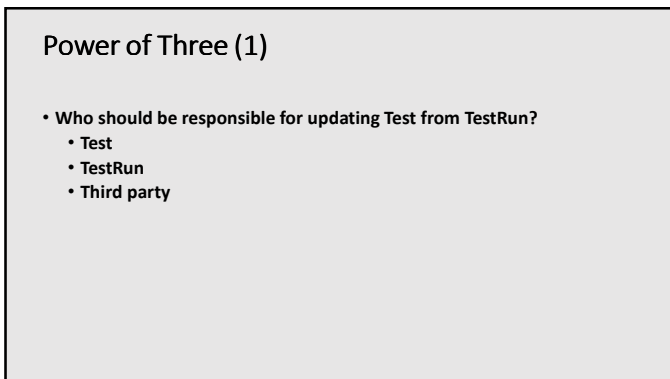
44



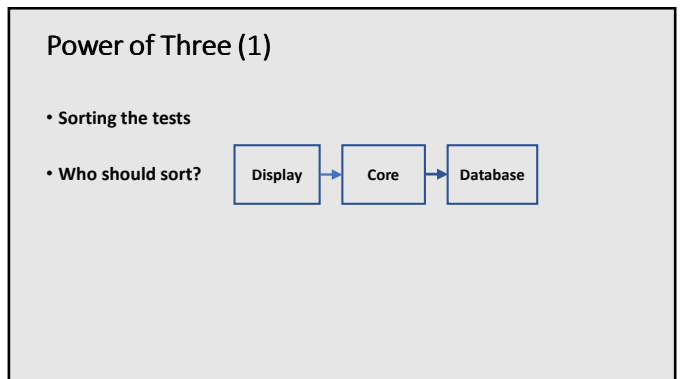
45



46



47



48

Power of Three (2)

- Filtering the test
- Who should filter?

```
graph LR; Display[Display] --> Core[Core]; Core --> Database[Database];
```

49

Code / Test

50

Main Flow Version Two (1)

Scenario: Run a test successfully

Given test exists

Issue ID	Name	Runner	Last Result
12345	Enter test result		Failure
...			
Date Last Run	Date Previous Result		
Never	Never		
...			
File Path	Comments		
EnterTest.feature			

51

Main Flow Version Two (2)

And value for runner is

Sam

And value for current date is

Oct 1, 2022, 12:30:01 AM

When test is selected

Issue ID 12345

And test is run

Result Success
Comments Works great

52

Main Flow Version Two (3)

Then test is now

Issue ID	Name	Runner	Last Result
12345	Enter test result	Sam	Success
...			
Date Last Run	Date Previous Result		
Oct 1, 2022, 12:30:01 AM	Never		
...			
File Path	Comments		
EnterTestResult.feature	Works great		

53

Other Flows

Scenario: Add a test

Scenario: Add a test with same identifier is not allowed

Scenario: Run a test multiple times

54

Background Run for Every Scenario

```
Background:
Given configuration values are:
  | Variable | Value |
  | rootFilePath | C:\Users\KenV1\...\TestRecorder\target |
Given file exists
  | File Path | Contents |
  | EnterTest.feature | Select test \n Run it \n Check result |
```

55

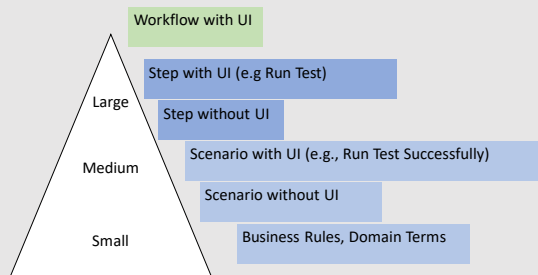
55

Test

56

56

Behavior Test Pyramid



57

Resources



ken.pugh@pugh-killeen.com
<https://www.linkedin.com/in/kenpugh/>
Twitter @kpugh
<http://acceptancetestdrivendevelopment.com> (<http://atdd.biz>)

59

59

Other Examples

- [A Coalesced View of Software Development](#)
- [Requirement Driven Development and Test-Driven Development](#)
- [A Behavior Perspective on Development](#)
- [The Auction Sniper – An ATDD/BDD Approach](#)
- [The Mars Rover Kata and BDD/ATDD](#)
- [Building Collaboration with Visible Tests](#)
- [Use Your Ubiquitous Language in Your Design](#)
- [The Gilded Rose Kata from a Gherkin Perspective](#)
- [A Dollar Kata](#)

58

Supplementary

60

Behavior

- Functional Behavior
- Display Behavior
 - Display rules
 - Appearance
- Connection between functional behavior and display behavior

61

61

Testing Approaches

- Parallel Testing
 - Use another instance (start up an environment) that starts the same state - for speeding up tests
 - Could set actual clock with different instances

62

62

Terms

- Validation – right format
- Verification – right existence

63

63

Guidelines

- Technology – is everything at the right level?
- Text or not to Text – outside is text, inside are objects
 - Avoid strings in executable code
- When you're abstract, be abstract all the way
 - Primitives are primitive – use them to construct objects
 - Strings are primitive
- Use names for constants in executable code
- Null on null – use "Null Objects"

64

64

Test Double Alternatives

- Access alternatives
 - Strategy -
 - in code – interface with alternatives and factory
 - Internal or external configuration
 - In build – use a different jar file
 - Is there a third?

65

65

Why ATDD/BDD?

- Streamline communication
- Decrease rework
- Increase productivity
- Raise customer satisfaction

66

66