Outside In Development

Marcus Ahnve
http://marcusahnve.org
marcus.ahnve@valtech.se
marcus@ahnve.com
@mahnve

valtech_
http://www.valtech.se

tl;dr
BDD/ATDD
Outside in development
There will be live coding
About me
Playing Agile Coach
Developed software professionally since 1996
Rubyist
About you
Developers?
ATDD/BDD/...?
Cucumber/Fitnesse/...?
RSpec/...?
Why automatic testing?
Make sure it works now
Make sure it works whenever
Why TDD?
Make sure it works now
Make sure it works whenever
Know when you're done
Why Outside In BDD?
Make sure it works now
Make sure it works whenever
Know when you're done
Build just what's needed
Design from functionality
Documentation - why
What is BDD?
"BDD is TDD done right"
"TDD means 'write the test first'. BDD takes this idea to a more general level: 'write the client first' (the outside)"
- Aslak Hellesøy
My BDD history
Java and Ruby developer
TestDox, Joe Walnes
Dan North
Dave Astels, Aslak Hellesøy, David Chelimsky
TestDox Code

```
public class AdditionTest {

  public void testOnePlusOneShouldBeTwo() {
    assertEqual(2, 1+1)
  }

  public void testTwoPlusThreeShouldBeFive() {
    assertEqual(5, 2+3)
  }
}
```

TestDox Output

```
Addition:

- one plus one should be two
- two plus three should be five
```

Coding By Example
Write an example explaining how we want the code to behave
Method Testing

```
class Klazz
class Klazz
  def method
    do_something
  end
end

class KlazzTest
  def setup
    @klazz = Klazz.new
  end
  def test_method
    result = @klazz.do_something
    assert_equal("expected", result)
  end
end
```

The Context Problem

```
class KlazzTest
  def setup
    @klazz = Klazz.new
  end
  def test_method
    result = @klazz.do_something
    assert_equal(:expected, result)
  end
  def test_another_context
    klazz = Klazz.new(:constructor_args)
    assert_equal(:expected_2, klazz.do_something)
  end
end
```

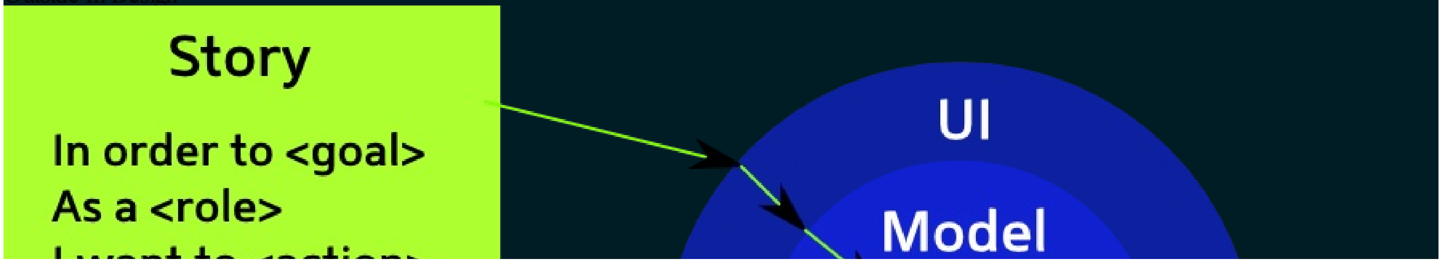
RSpec

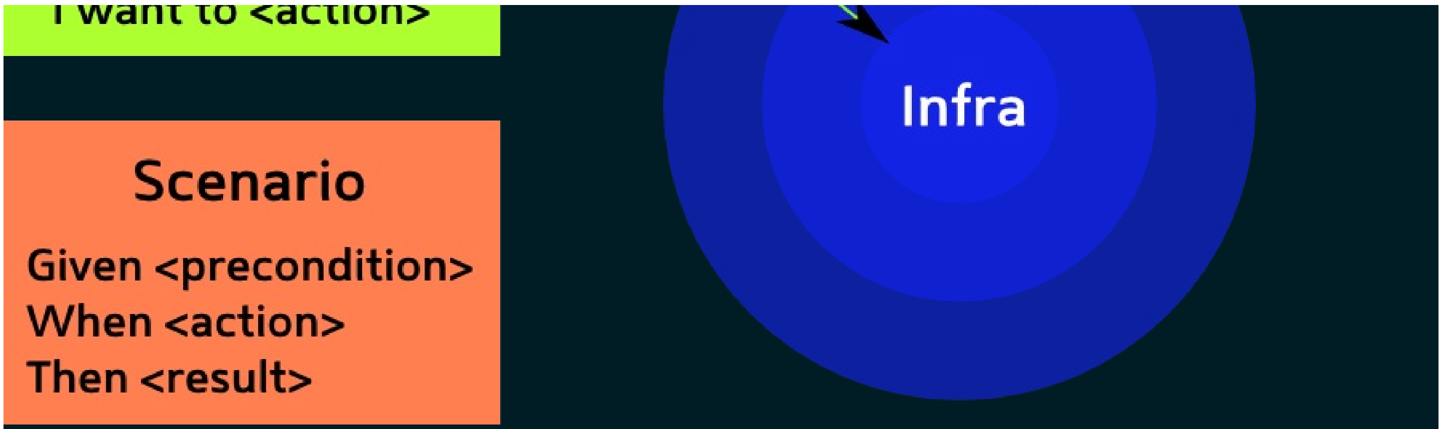
```
describe Klazz do

  before :all do
    @klazz = Klazz.new
  end

  it 'should do something' do
    @klazz.do_something.should == 'expected'
  end

  context "other" do

    before :all do
      @klazz = Klazz.new(:constructor_args)
    end

    it 'should do something else' do
      @klazz.do_something.should == 'something else'
    end
  end
end
```

The Problem With Classic OO

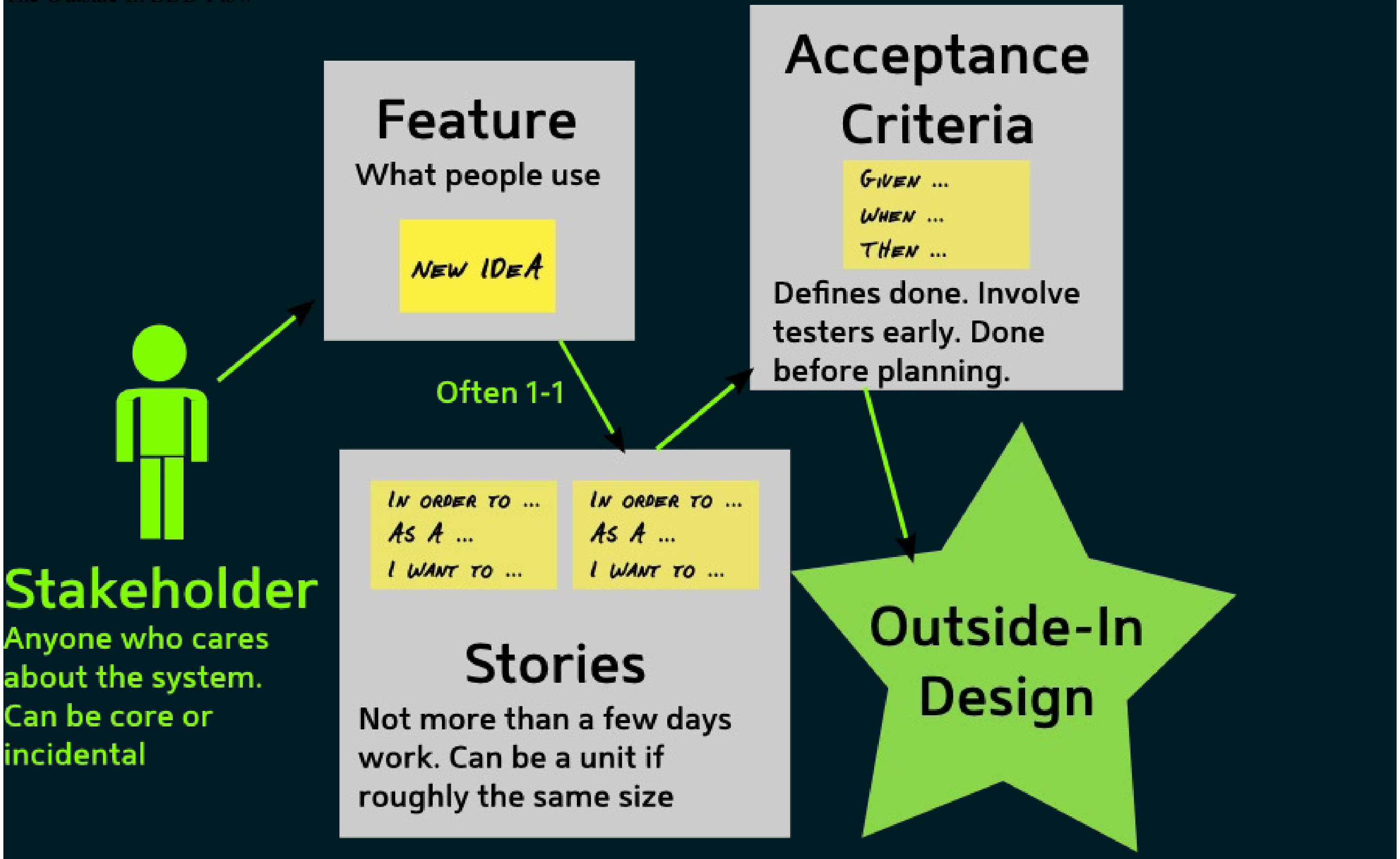
Focuses on internals
Very often not what the UI wants
Inside out

Outside In Design





The Outside In BDD Flow



Cucumber

```
Feature: Addition

  In order to get correct sums
  As a user I can add numbers

  Scenario: 1+1
    Given I want to add numbers
    When I add 1+1
    Then I the answer should be 2
```

Don't Focus On GUI Details

```
Feature: Register Sale

  Given I an editor
  When I am at "/"
  And I click "Admin"
  And I click "New Post"
  And I enter "New Post" in "Title" field
  And I enter "Blah blah blah" in "Body" field
  And I click "Save"
  Then I should see "Message saved"
  And the post should be saved in the database
  And the post should be saved in the database
  And ...
```

Focus On business Logic

```
Feature: Register Sale

  Given I an editor
  When I write a new post
  Then the post should be available in latest news list
  And I should receive a feedback message
```

Hey! Where is the code?
Demo
Thank You!
http://marcusahnve.org
marcus.ahnve@valtech.se
marcus@ahnve.com
@mahnve
valtech_
http://www.valtech.se
valtech_