let's move
the java world

Behaviour Driven Development with Cucumber for Java

Thomas Sundberg

Thomas Sundberg

Developer for more than 20 years

Masters degree in Computer Science from the Royal Institute of Technology, KTH, in Stockholm, Sweden

I write computer programs

@thomassundberg
tsu@kth.se
http://thomassundberg.wordpress.com

Goal

- Introduce Behaviour Driven Development - BDD
- Introduce Cucumber for Java

Disposition

- Why – Theory
- How – Live coding
- Recap – the Why and How again

Tools are useless

- If you don't know why and how to use them

Why

- Communication

Where are we on the map?

Business

Technology

Domain Driven Development - DDD

Test Driven Development - TDD

Behaviour Driven Development - BDD

Best of two worlds

- Domain Driven Design - DDD
  - Common understanding
  - Common language
- Test Driven Development - TDD
  - Good technical practices
  - Small steps
- Combined → Behaviour Driven Development

Test automation

- Test Driven Development, TDD – Develop it the right way
- Behaviour Driven Development, BDD – Develop the right thing

- Behaviour Driven Development

- ***Behaviour*** Driven Development

Black box

Three core principles

- Business and Technology should refer to the same system in the same way
- Any system should have an identified, verifiable value
- Up-front analysis, design and planning all have a diminishing return

Common language

- Used by all involved
  - Customer
  - Developers
  - Others?

Verifiable value

- Protect revenue
- Increase revenue
- Manage cost
- Increase brand value
- Make the product remarkable
- Provide more value to your customer

Big design up-front

- No big design up-front
- The further you plan ahead, the less accurate you will be

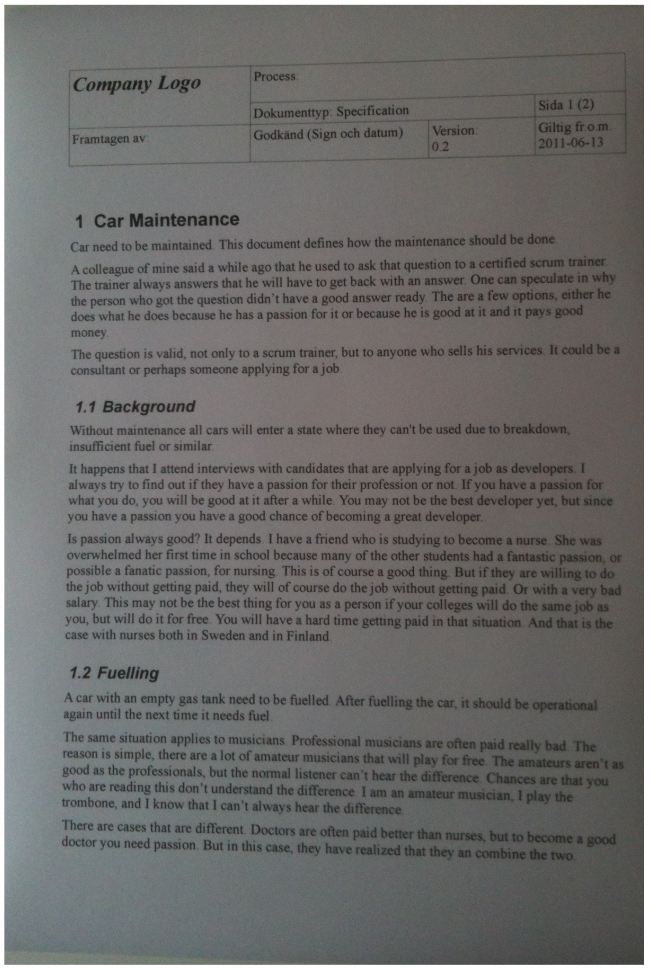
Three questions

- What is the most important thing the system should do?
- What is the next most important thing the system doesn't do yet?
- If we were to switch off the system, where and what would be the biggest impact?

Vague answers?

- 5Y

Time to create a specification



Company Logo	Process:		
	Dokumenttyp: Specification		Sida 1 (2)
Framtagen av:	Godkänd (Sign och datum)	Version: 0.2	Giltig fr.o.m. 2011-06-13

1 Car Maintenance

Car need to be maintained. This document defines how the maintenance should be done.

A colleague of mine said a while ago that he used to ask that question to a certified scrum trainer. The trainer always answers that he will have to get back with an answer. One can speculate in why the person who got the question didn't have a good answer ready. The are a few options, either he does what he does because he has a passion for it or because he is good at it and it pays good money.

The question is valid, not only to a scrum trainer, but to anyone who sells his services. It could be a consultant or perhaps someone applying for a job.

1.1 Background

Without maintenance all cars will enter a state where they can't be used due to breakdown, insufficient fuel or similar.

It happens that I attend interviews with candidates that are applying for a job as developers. I always try to find out if they have a passion for their profession or not. If you have a passion for what you do, you will be good at it after a while. You may not be the best developer yet, but since you have a passion you have a good chance of becoming a great developer.

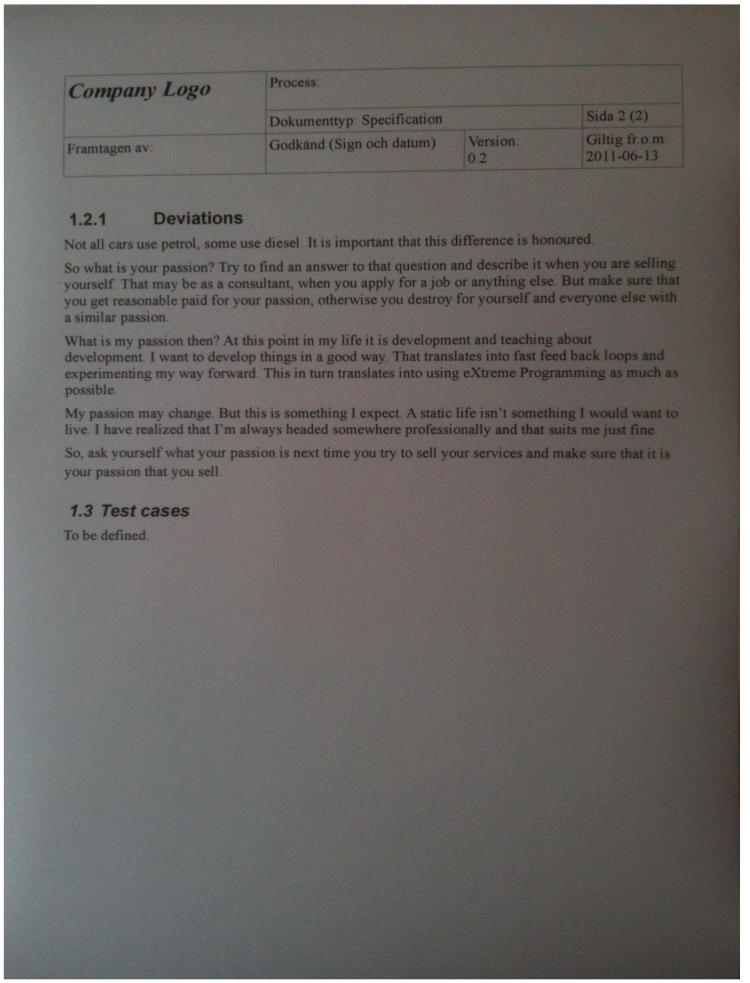
Is passion always good? It depends. I have a friend who is studying to become a nurse. She was overwhelmed her first time in school because many of the other students had a fantastic passion, or possible a fanatic passion, for nursing. This is of course a good thing. But if they are willing to do the job without getting paid, they will of course do the job without getting paid. Or with a very bad salary. This may not be the best thing for you as a person if your colleges will do the same job as you, but will do it for free. You will have a hard time getting paid in that situation. And that is the case with nurses both in Sweden and in Finland.

1.2 Fuelling

A car with an empty gas tank need to be fuelled. After fuelling the car, it should be operational again until the next time it needs fuel.

The same situation applies to musicians. Professional musicians are often paid really bad. The reason is simple, there are a lot of amateur musicians that will play for free. The amateurs aren't as good as the professionals, but the normal listener can't hear the difference. Chances are that you who are reading this don't understand the difference. I am an amateur musician, I play the trombone, and I know that I can't always hear the difference.

There are cases that are different. Doctors are often paid better than nurses, but to become a good doctor you need passion. But in this case, they have realized that they an combine the two.



Company Logo	Process:		
	Dokumenttyp: Specification		Sida 2 (2)
Framtagen av:	Godkänd (Sign och datum)	Version: 0.2	Giltig fr.o.m. 2011-06-13

1.2.1 Deviations

Not all cars use petrol, some use diesel. It is important that this difference is honoured.

So what is your passion? Try to find an answer to that question and describe it when you are selling yourself. That may be as a consultant, when you apply for a job or anything else. But make sure that you get reasonable paid for your passion, otherwise you destroy for yourself and everyone else with a similar passion.

What is my passion then? At this point in my life it is development and teaching about development. I want to develop things in a good way. That translates into fast feed back loops and experimenting my way forward. This in turn translates into using eXtreme Programming as much as possible.

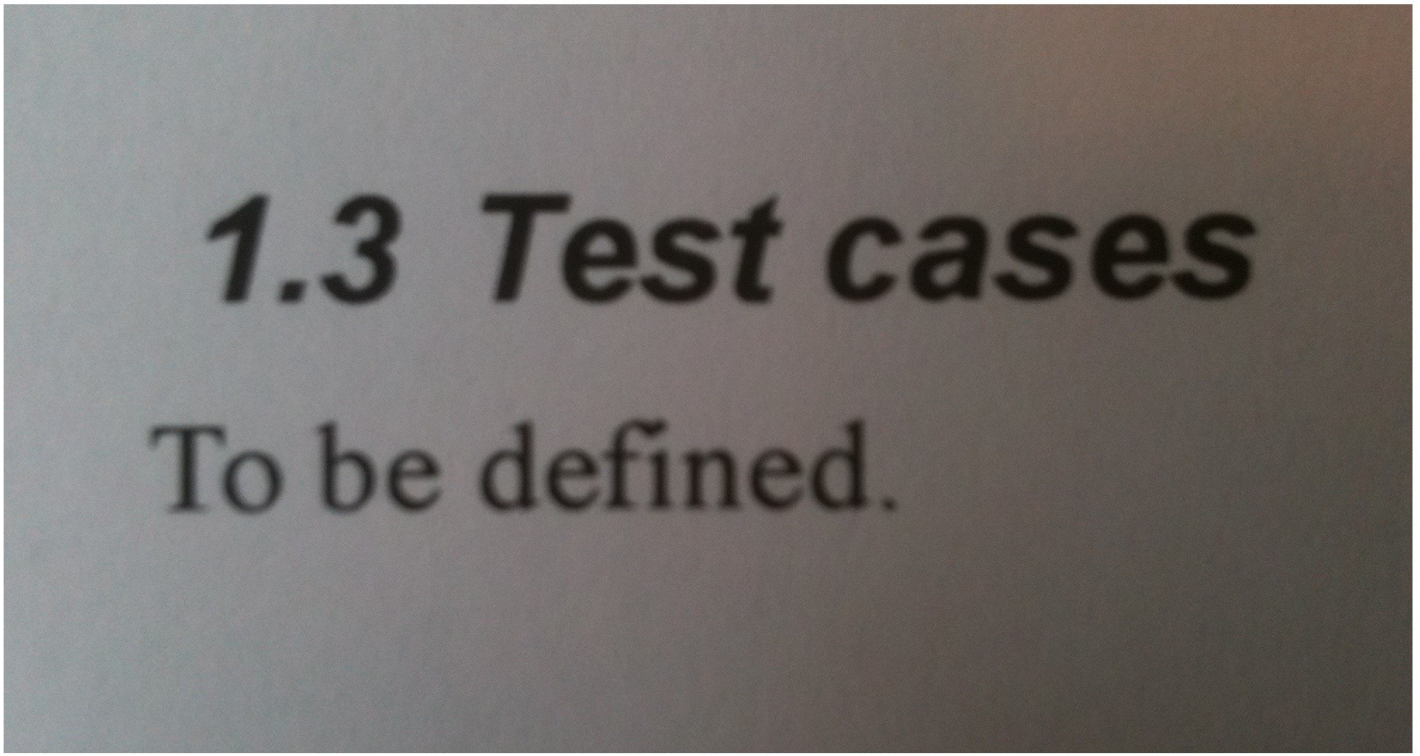
My passion may change. But this is something I expect. A static life isn't something I would want to live. I have realized that I'm always headed somewhere professionally and that suits me just fine.

So, ask yourself what your passion is next time you try to sell your services and make sure that it is your passion that you sell.

1.3 Test cases

To be defined.

Time to create a specification



When are we done?

Interpret and translate

- We need to read the spec.
- Understand it
- Translate it into code

- Parse and execute it?
- No, not all specs are created equal

Solution

- Write specifications so
  - Translations isn't needed
  - It is possible to execute them

Create a concrete example

Mark Twain:

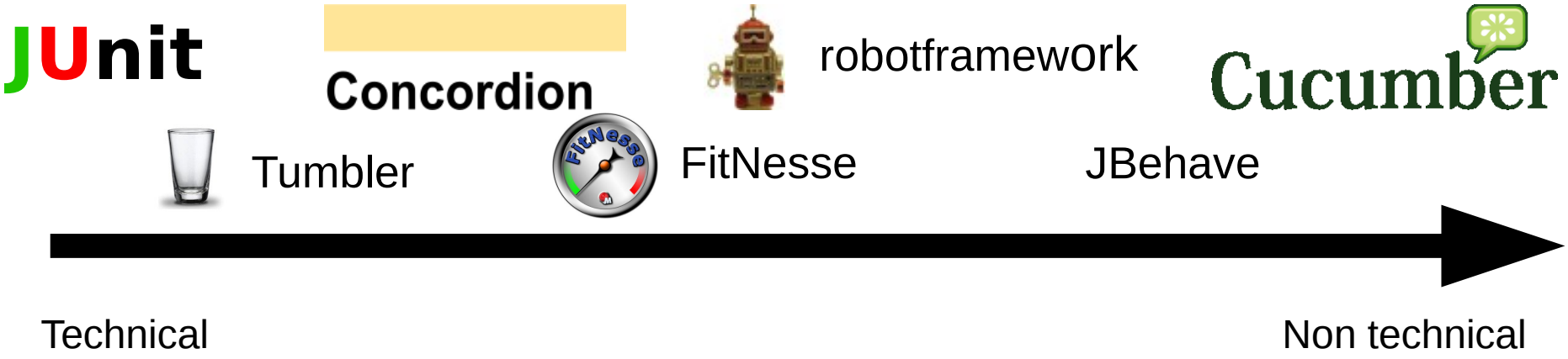
There is nothing so annoying as a good example

Format

- Given – Setup the system
- When – Change the state of the system
- Then – Verify the new state

The right tool for the job

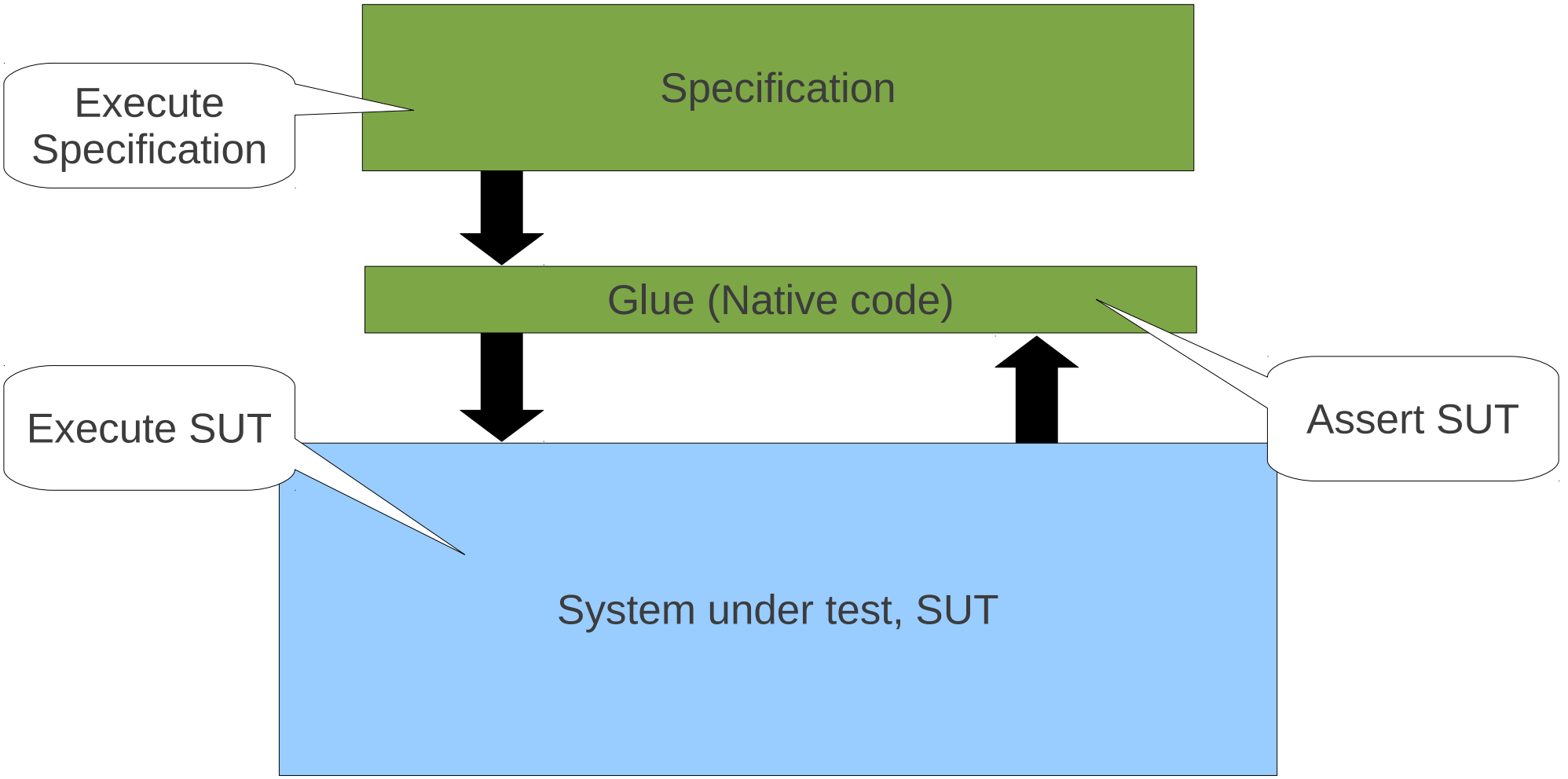
Tools


JUnit
Concordion
robotframework
Cucumber
Tumbler
FitNesse
JBehave
Technical
Non technical

Audience

- Readers
  - Customers
  - Developers
- Maintainers
  - Product owner
  - Developers

Pattern


Specification
Execute Specification
Glue (Native code)
Execute SUT
Assert SUT
System under test, SUT

Why Cucumber

- It is one of the least technical tools
- It descends from RSpec
- It is a very active open source project
- It supports a variety of languages

A concrete example

```
Feature: Daily car maintenance
  As a car owner
  I want to be able to drive my car
  So I can get where I want

Scenario: Fuelling
  Given a car with 10 litres of fuel in the tank
  When you fill it with 50 litres of fuel
  Then the tank contains 60 litres
```

Advantages

- The format is
  - Easy to read
  - Easy to understand
  - Easy to discuss
  - Easy to parse

Parse and execute

- A parser is developed
- Add native code to implement the steps
  - Setup – Given
  - Execute – When
  - Assert - Then

How

- Outline of the example
- Live coding

Example

- A Maven project
- One feature
- Simple model
- Use a Continuous Integration server
- Extend to a web application

Enough slides

Coding...

Small example

- All large systems consists of small pieces
- You can only view a small portion of a system at one time

  ~30 – 50 loc

http://www.casualmiracles.com/2010/02/21/large-systems/

Workflow

1. Describe the behaviour in plain text

```
Feature: Addition
  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers

  Scenario: Add two numbers
    Given I have entered 50 into the calculator
    And I have entered 70 into the calculator
    When I press add
    Then the result should be 120 on the screen
```

2. Write a step definition in Ruby

```
Given /I have entered (.*) into the calculator/ do |n|
  calculator = Calculator.new
  calculator.push(n.to_i)
end
```

2. Write a step definition in ~~Ruby~~ Java

```
Given /I have entered (.*) into the calculator/ do |n|
  calculator = Calculator.new
  calculator.push(n.to_i)
end
```

3. Run it and watch it fail

```
$ cucumber features/addition.feature
Feature: Addition  # features/addition.feature
  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers
  Scenario: Add two numbers                                  # features/additi
    Given I have entered 50 into the calculator              # features/step_d
    And I have entered 70 into the calculator                # features/step_d
    When I press add                                         # features/additi
    Then the result should be 120 on the screen              # features/additi
```

4. Write code to make the step pass

```
class Calculator
  def push(n)
    @args ||= []
    @args << n
  end
end
```

5. Run it again and see the step pass

```
$ cucumber features/addition.feature
Feature: Addition  # features/addition.feature
  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers
  Scenario: Add two numbers                                  # features/additi
    Given I have entered 50 into the calculator              # features/step_d
    And I have entered 70 into the calculator                # features/step_d
    When I press add                                         # features/additi
    Then the result should be 120 on the screen              # features/additi
```

6. Repeat step 2 – 5 until green like a Cuke

```
$ cucumber features/addition.feature
Feature: Addition  # features/addition.feature
  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers
  Scenario: Add two numbers                                  # features/additi
    Given I have entered 50 into the calculator              # features/step_d
    And I have entered 70 into the calculator                # features/step_d
    When I press add                                         # features/step_d
    Then the result should be 120 on the screen              # features/step_d
```

7. Repeat step 1 – 6 until the money runs out

1. Describe the behaviour in plain text
2. Write a step definition
3. Run it and watch it fail
4. Write code to make the step pass
5. Run it again and see the step pass
6. Repeat step 2 – 5 until green like a Cuke

Why is Cucumber better

- Compared to well known tools like JUnit?
- It isn't, it's just easier to read for non coders
- The readability has increased
- Communication is easier

Is this a new Silver bullet?

- Absolutely not
- Will increase our chances to build the right thing
- A fool with a tool is still a fool

Benefits

- Less unnecessary work
- Better regression tests
- Less speculative work
- Less re-work

Maintenance

- Maintained together
  - Specs
  - Tests
  - Code
- Maven
- Continuous Integration, CI

Tips

- Always use the a common language
- Don't over specify – start easy
- Never specify implementation details

Do not focus on tools

- They will never solve the problem
- A fool with a tool is still a fool

Resources

- Cucumber - http://cukes.info/
- Selenium - http://seleniumhq.org/
- Maven - http://maven.apache.org/
- Jenkins - http://jenkins-ci.org/
- Blog - http://thomassundberg.wordpress.com/

Thomas Sundberg

Developer for more than 20 years

Masters degree in Computer Science from the Royal Institute of Technology, KTH, in Stockholm, Sweden

I write computer programs

@thomassundberg
tsu@kth.se
http://thomassundberg.wordpress.com