Definition

-of-

Done

How do we know we're Done?

How do we know we're Done?

it works!

How do we know we're Done?

it works!

it's quality!

How do we know we're Done?

it works!

it's quality!

it's tested!

How do we know we're Done?

it works!

it's quality!

it's tested!

it's in production!

How do we know we're Done?

it works!

it's quality!

it's tested!

it's in production!

it's the deadline!

How do we know we're Done?

it works!

it's quality!

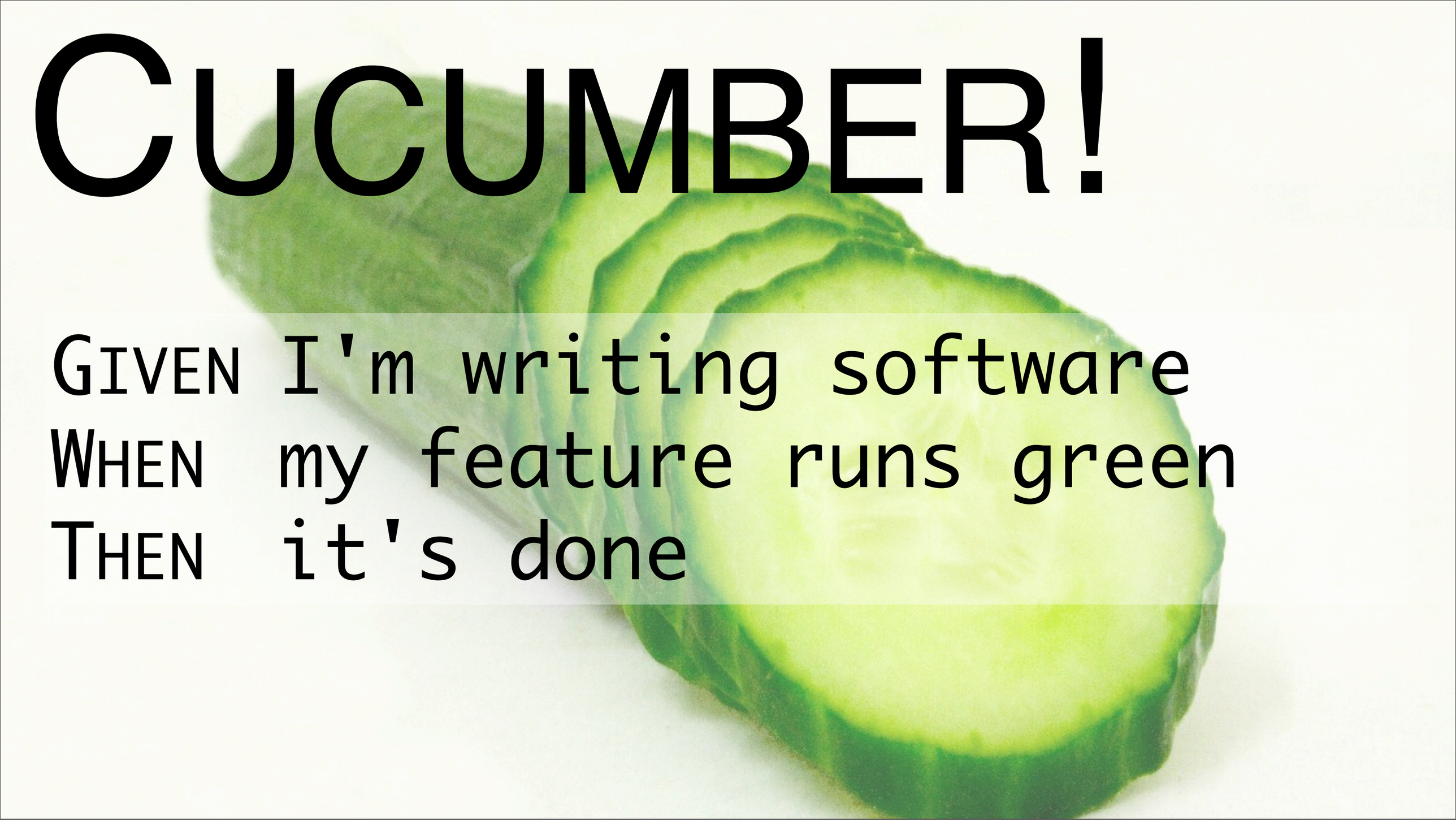
it's tested!

it's in production!

it's the deadline!

its funding ran out!

Cucumber!



```
Given I'm writing software
When  my feature runs green
Then  it's done
```

THEY CAN'T HIDE BEHIND AMBIGUITY

YOU CAN'T HIDE BEHIND AMBIGUITY

for example

```
Feature: using a calculator
```

```
Feature: using a calculator

  Background:
```

```
Feature: using a calculator

  Background:
    Given the calculator is open
```

```
Feature: using a calculator

  Background:
    Given the calculator is open

  Scenario: adding two numbers
```

```
Feature: using a calculator

  Background:
    Given the calculator is open
```

```
  Scenario: adding two numbers
    Given I want to add two numbers
```

```
Feature: using a calculator

  Background:
    Given the calculator is open

  Scenario: adding two numbers
    Given I want to add two numbers
    When I perform the calculation
```

```
Feature: using a calculator

  Background:
    Given the calculator is open

  Scenario: adding two numbers
    Given I want to add two numbers
    When I perform the calculation
    Then I should see my two numbers added
```

@wip it!

```
Feature: using a calculator

  Background:
    Given the calculator is open

  Scenario: adding two numbers
    Given I want to add two numbers
    When I perform the calculation
    Then I should see my two numbers added
```

```
@wip
Feature: using a calculator

  Background:
    Given the calculator is open
```

```
  Scenario: adding two numbers
    Given I want to add two numbers
    When I perform the calculation
    Then I should see my two numbers added
```

Start the server:

```
$ bundle exec ruby server.rb
```

Start the server:

```
$ bundle exec ruby server.rb
```

Run the work-in-progress cukes:

```
$ bundle exec cucumber --tags @wip --wip
```

```
@wip
Feature: using a calculator

Scenario: adding two numbers

  Given I want to add two numbers
  When I perform the calculation
  Then I should see my two numbers added # features/calculator.feature:7

You can implement step definitions for undefined steps with these snippets:

Given /^I want to add two numbers$/ do
  pending # express the regexp above with the code you wish you had
end

When /^I perform the calculation$/ do
  pending # express the regexp above with the code you wish you had
end

Then /^I should see my two numbers added$/ do
  pending # express the regexp above with the code you wish you had
end

1 scenario (1 undefined)
3 steps (3 undefined)
0m0.004s

The --wip switch was used, so the failures were expected. All is good.
```

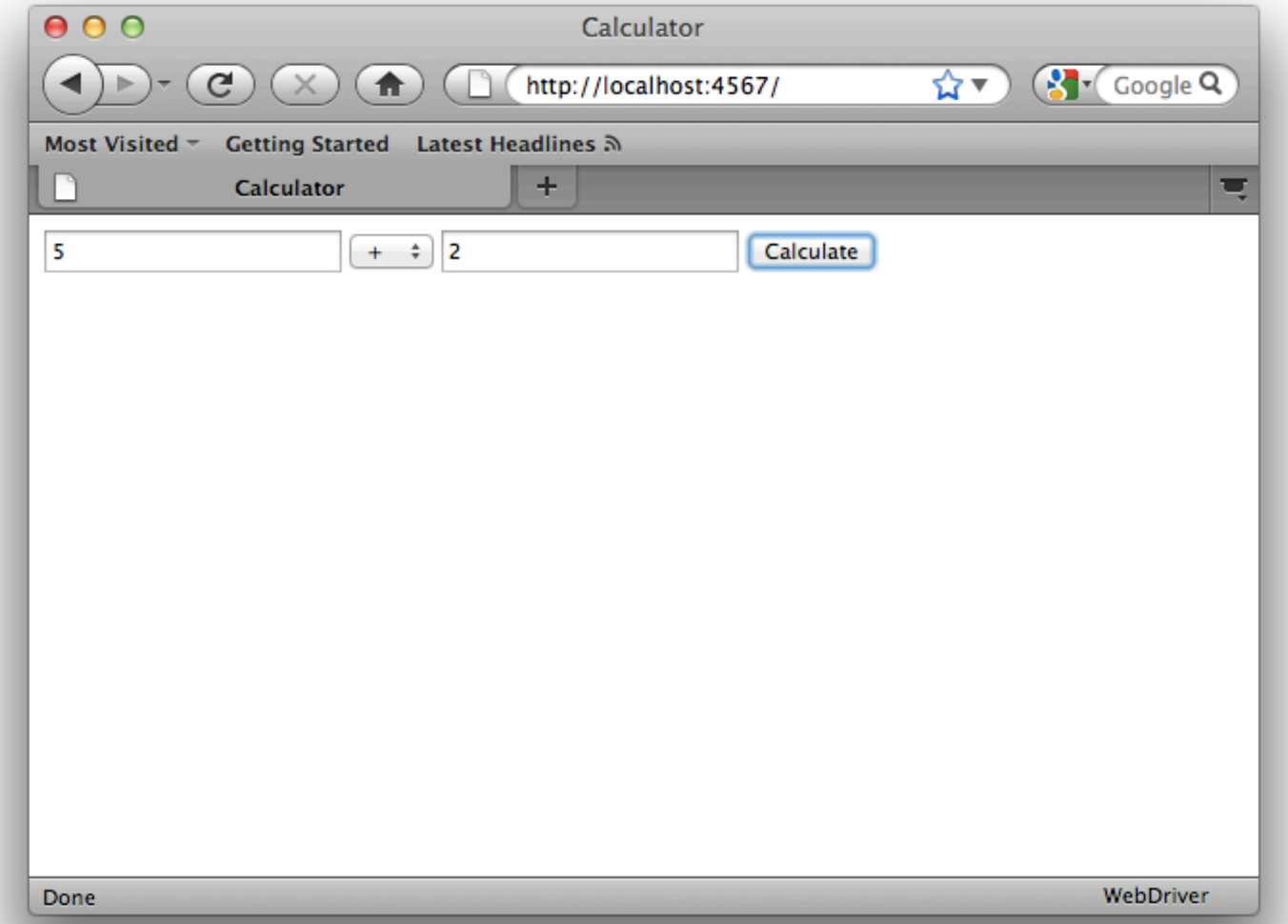
automate your steps in Ruby

```
Given /^the calculator is open$/ do
  visit '/'
end
```

```
Given /^the calculator is open$/ do
  visit '/'
end

Given /^I want to add two numbers$/ do
  fill_in "leftOperand", :with => 5
  select '+', :from => 'operator'
  fill_in "rightOperand", :with => 2
end
```

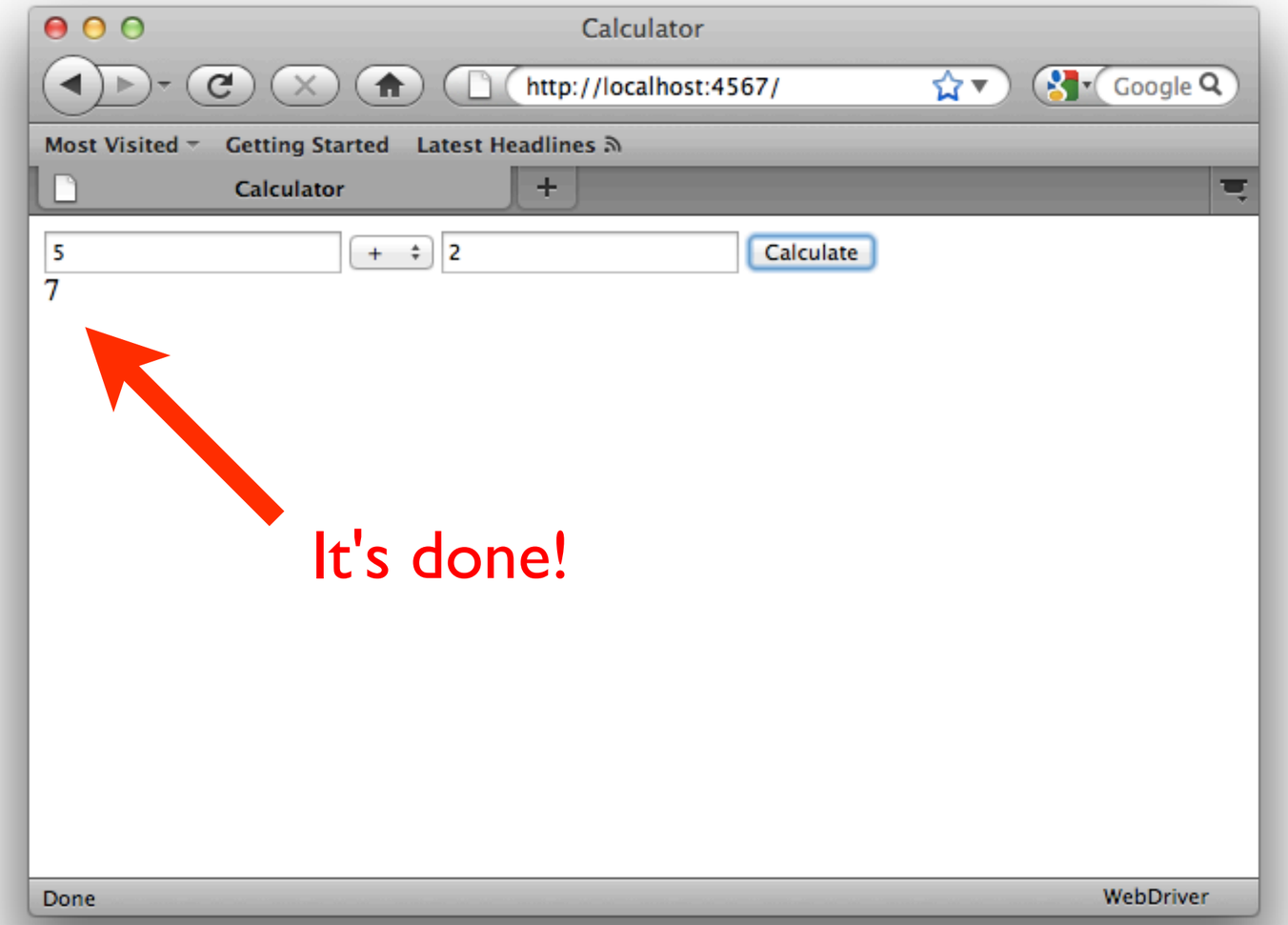
```
Given /^the calculator is open$/ do
  visit '/'
end

Given /^I want to add two numbers$/ do
  fill_in "leftOperand", :with => 5
  select '+', :from => 'operator'
  fill_in "rightOperand", :with => 2
end

When /^I perform the calculation$/ do
  click_button 'Calculate'
end
```

```ruby
Given /^the calculator is open$/ do
  visit '/'
end

Given /^I want to add two numbers$/ do
  fill_in "leftOperand", :with => 5
  select '+', :from => 'operator'
  fill_in "rightOperand", :with => 2
end

When /^I perform the calculation$/ do
  click_button 'Calculate'
end

Then /^I should see my two numbers added$/ do
  find('.result').should have_content 7
end
```


Calculator
http://localhost:4567/
Google
Most Visited
Getting Started
Latest Headlines
Calculator
5
+
2
Calculate
Done
WebDriver

```
@wip
Feature: using a calculator

Background:
  Given the calculator is open

Scenario: adding two numbers
  Given I want to add two numbers
  When I perform the calculation
  Then I should see my two numbers added
    expected there to be content "7" in "" (RSpec::Expectations::ExpectationNotMetErr

Failing Scenarios:
  cucumber features/calculator.feature:7

1 scenario (1 failed)
4 steps (1 failed, 3 passed)
0m6.380s
The --wip switch was used, so the failures were expected. All is good.
```

hack hack hack


Calculator
http://localhost:4567/
Google
Most Visited
Getting Started
Latest Headlines
Calculator
5
+
2
Calculate
7
It's done!
Done
WebDriver

```
@wip
Feature: using a calculator

Background:
  Given the calculator is open

Scenario: adding two numbers

  Given I want to add two numbers
  When I perform the calculation
  Then I should see my two numbers added
```

```
1 scenario (1 passed)
4 steps (4 passed)
0m5.199s

The --wip switch was used, so I didn't expect anything to pass.

These scenarios passed:

(::) passed scenarios (::)
features/calculator.feature:7:in `Scenario: adding two numbers'
```

un-@wip it!

```
Feature: using a calculator

  Background:
    Given the calculator is open

  Scenario: adding two numbers
    Given I want to add two numbers
    When I perform the calculation
    Then I should see my two numbers added
```

Run the "done" regression cukes:

```
$ bundle exec cucumber --tags ~@wip
```

```
Feature: using a calculator

Background:
  Given the calculator is open

Scenario: adding two numbers

  Given I want to add two numbers
  When I perform the calculation
  Then I should see my two numbers added

1 scenario (1 passed)
4 steps (4 passed)
0m4.228s
```

Exercises

1. Add a failing scenario for subtraction, then implement it!

2. Next, try adding division!

3. Try adding a big table bunch of examples and edge cases using a Cucumber Scenario Outline
   (details here: http://is.gd/cukeoutline)