Testing C# and ASP.net using Ruby

@Ben_Hall
Ben@BenHall.me.uk
Blog.BenHall.me.uk

Meerkatalyst

London based C# MVP
Web Developer @ 7digital.com

Working on a number of Open Source Projects

Co-Author of
Testing ASP.net Web Applications
http://www.testingaspnet.com

How we can apply Ruby testing techniques to C#

- In mindset
- In practice

1| Object Level testing
2| Systems testing

http://www.flickr.com/photos/atomicpuppy/2132073976/

VALUE

Realised Ruby doesn’t have as many problems as .net

Natural Language

Test Driven Development

TDD

I love writing tests upfront

Not sure about implementation

According to various TDD books \ blogs

TDD uptake still low

At least within .net

Help with an initial design but what about long term?

“Unit”


TRANSFORMERS
©2002 Hasbro, Inc. All rights reserved. www.transformers.com

Behaviour Driven Development

BDD

Shift focus

Final Intent

Code Coverage

Context \ Specification

RSpec

//www.flickr.com/photos/dod

```
describe MyObject, “In this situation” do
   it “should do this” do
   end

   it “should also do this” do
   end

   it “should not do this” do
   end
end
```


IronRuby

C#

```
public class ProductController : Controller
{
  IStoreService _service;

  public ProductController(IStoreService service)
  {
    _service = service;
  }
}
```

Ruby (via IronRuby)

```
p = ProductController.new(nil)
```

Thanks to TekPub for Kona sample

Define the context

C#

```
public ActionResult Index(int? id)
{
  ProductListViewModel model = null;
  if (id.HasValue)
    model =
    _service.GetHomeModel(id.Value);
  else
    return RedirectToAction("Index",
    "Home");

    return View(model);
}
```

Ruby (via IronRuby)

```
describe ProductController, “Product homepage
    requested" do
 context "when Category ID is empty" do
 end
end
```

Thanks to TekPub for Kona sample

Create the context

```
$: << '../../Kona/bin/'
require 'Kona.dll'
Include Kona::Controllers

describe ProductController, "Product homepage requested" do
  context "when Category ID is empty" do
  end
end
```

Thanks to TekPub for Kona sample

Create the context

```
require '../spec_helper‘

describe ProductController, “Product homepage requested" do
  context "when Category ID is empty" do
  end
end
```

Thanks to TekPub for Kona sample

Create the context

```
require '../spec_helper‘

describe ProductController, “Product homepage requested" do
  context "when Category ID is empty" do
    before(:each) do
      controller = ProductController.new nil

      category_id = nil
      @result = controller.Index category_id
    end
  end
end
```

Thanks to TekPub for Kona sample

Define the spec

```
require '../spec_helper‘

describe ProductController, “Product homepage requested" do
  context “with empty Category ID" do
     before(:each) do ... End

    it "should redirect to main landing page" do
        @result.route_values[‘controller'].should == ‘Index’
       @result.route_values[‘home'].should == ‘Home’
    end
end
```

Thanks to TekPub for Kona sample

Stubbing via caricature

```
context “with validCategory ID" do
  Before(:each)
     view_model = product_list('Test Category')
     service = isolate Services::IStoreService
     service.when_receiving(:get_home_model).with(valid_category_id).return(view_model)
     controller = ProductController.new service
     @result = controller.Index valid_category_id
  end

  it “returns a view model” do
    @result.view_model.should_not == nil
  end
  it “should return name of selected category” do
    @result.view_model.selected_category.name.should == 'Test Category‘
  end
end
```

Thanks to TekPub for Kona sample

So what?

Limited by CLR

IStoreService and NHibernate

```
public class StoreService : IStoreService
{
  ISession _session;
  public StoreService(ISession session) {
    _session = session;
  }

  public ProductListViewModel GetHomeModel(int categoryId)
  {
    var categories = _session.CreateCriteria<Category>().Future<Category>();
    result.Categories = categories;
    return result;
  }
}
```

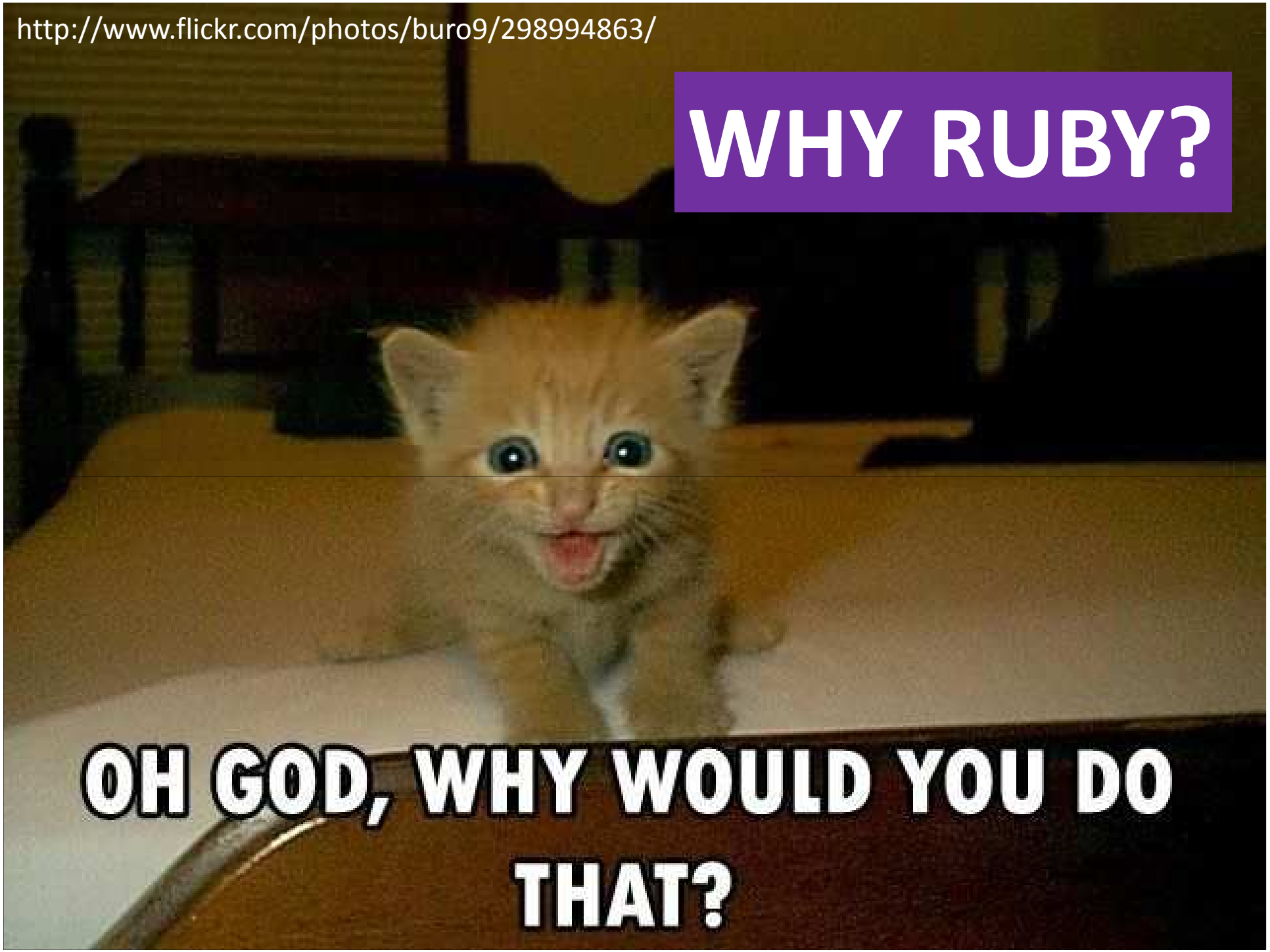
Thanks to TekPub for Kona sample

Tests and Databases

```
describe Services::StoreService do
  before(:all) do
    session = NHibernate::create_session
    NHibernate::insert_category session
    Services::StoreService.new session
    @model = service.get_home_model 0
  end

  it "should return constructed object" do
    @model.should_not be_nil
  end

  it "should return all categories from database" do
    @model.Categories.to_a.Count.should == 1
  end
end
```

Thanks to TekPub for Kona sample

```
describe CheckoutController, “Purchasing products from basket” do
  let(:payment_service) { create_payment_service_stub}
  let(:controller) {CheckoutController.new(payment_service)}

  context “with empty basket“
      before(:all)...end
       it_should_behave_like “user needs to be logged in”
       it “displays message relating 0 items"
  end

  context “with valid products“ do
    before(:all)...end
    context “payment” do
      it “should reject an invalid payment type”
      it “should reject a card with invalid CC number”
      it “should pass details to payment service with total if details are valid”
    end
    context “confirmation” do
      it “should return a confirmation of processing payment if valid”
      it “should return an error if payment is invalid”
      it “should empty the basket”
    end
  end
end
```


http://www.flickr.com/photos/buro9/298994863/
WHY RUBY?
OH GOD, WHY WOULD YOU DO THAT?

```
[Subject(typeof(HomeController))]
public class when_the_navigation_controller_is_asked_for_the_header :
                                                                    specification_for_navigation_controller
{
    static ActionResult result;

    Establish context = () => identity_tasks.Stub(i => i.IsSignedIn()).Return(true);

    Because of = () => result = subject.Menu();

    It should_ask_the_identity_tasks_if_the_user_is_signed_in = () =>
        identity_tasks.AssertWasCalled(x => x.IsSignedIn());

    It should_return_the_default_view = () => result.ShouldBeAView().And().ViewName.ShouldBeEmpty();

    It should_not_use_a_master_page = () => result.ShouldBeAView().And().MasterName.ShouldBeEmpty();

    It should_set_the_view_model_property_to_a_new_menu_view_model = () =>
        result.Model<MenuViewModel>().ShouldNotBeNull();

    It should_set_the_properties_of_the_view_model_correctly = () =>
        result.Model<MenuViewModel>().IsLoggedIn.ShouldBeTrue();
  }
```

() =>

```
describe HomeController, “When the navigation controller is asked for the header” do
  before(:all) do
    @identity_tasks.Stub(i => i.IsSignedIn()).Return(true);
    @result = subject.Menu();
  end

  it “should ask the identity tasks if the user is signed in” do
    @identity_tasks.did_receive(:is_signed_in) .should be_successful
  end

  it “should return the default view” do
    @result.should be_view
    @result.view_name.should be_empty
  end

  it “should not use a master page” do
    @result.should be_view
    @result.master_name.should be_empty
  end

 it “should set the view model property to a new menu view model” do
   @result.should be_view
   @result.model.should_not be_nil
 end

 it “should set the properties of the view model correctly” do
   @result.model.is_logged_in.should be_true
 end
end
```

You can make C# readable

But it’s hard

Consider your test suite containing > 500 tests

Each test matters.

But each problem hurts more

Share specs

```
share_examples_for "unauthorized user" do
  it 'sets an-error message' do
    result.view_model.error_message.should == "Sorry, you need to login to access."
  end

  it 'redirects to the login page' do
    result.view_name.should
  end
end
```

```
describe CheckoutController do
  it_should_behave_like “unauthorized user“

describe AccountController
  it_should_behave_like “unauthorized user“
```

Include additional functionality

```
require ‘nhibernate_helpers’
require ‘product_creation_extensions’
require  ‘customer_creation_extensions’
```

model.Categories.to_a.Count.should == 1

model.Categories.should have(1)

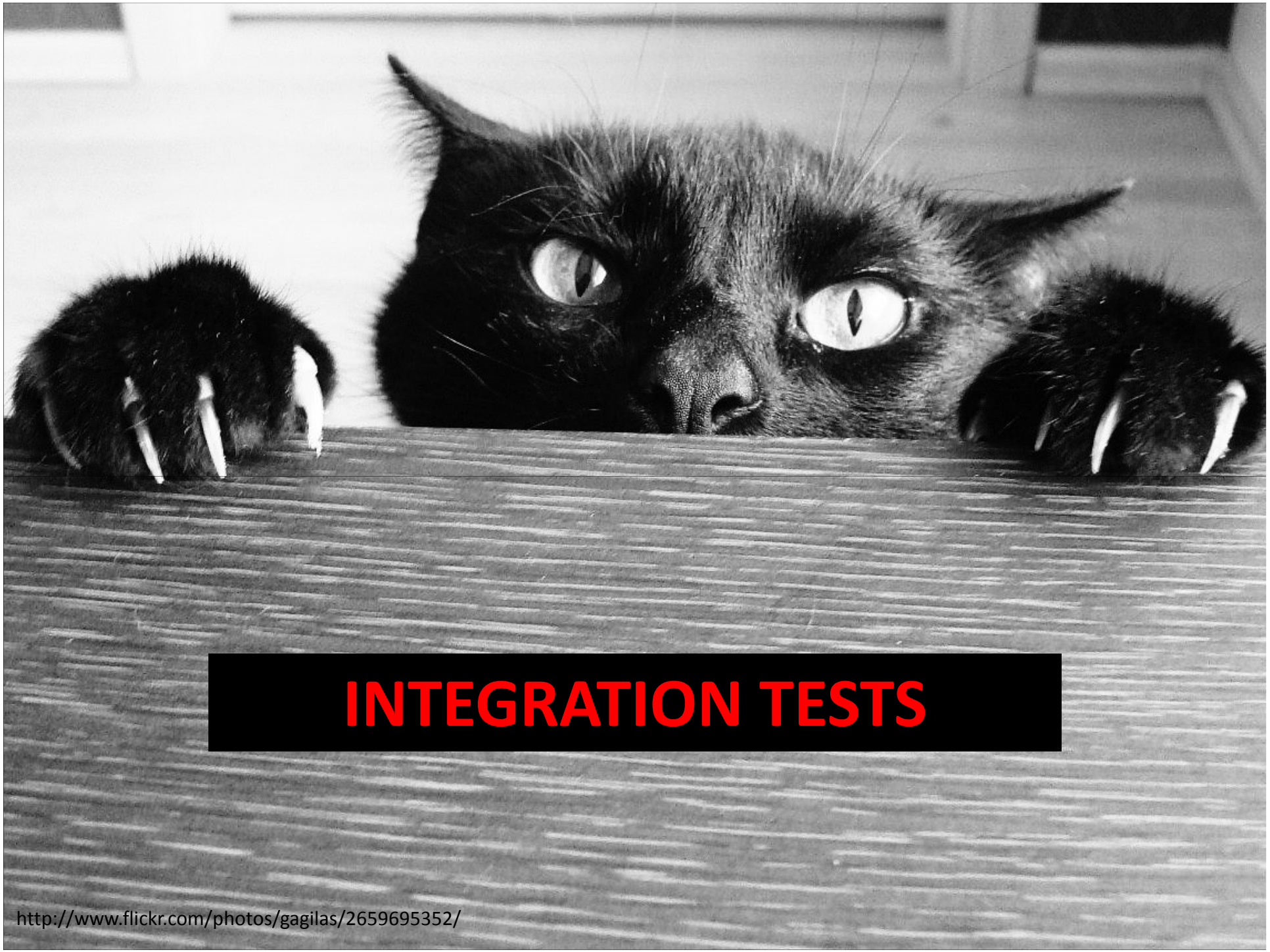
Duck Typing FTW!

```
module Matchers
  class CountCheck
    def initialize(expected)
      @expected = expected
    end

    def matches?(actual)
       actual.to_a.Count() == @expected
    end
  end

  def have(expected)
    CountCheck.new(expected)
  end
end
```

Dynamically create specs

```
def find_products(category)
   Product.Find(:all, :conditions => “category #{category}”)
end

describe ‘products should be able to be added to basket' do
 before(:all) do
   @basket = Basket.new
 end

 find_products(‘Boots’).each do |p|
   it “can add #{p.product_name} to basket" do
     @basket.add_item p.id
     @basket.should contain(p)
   end
 end
end
```

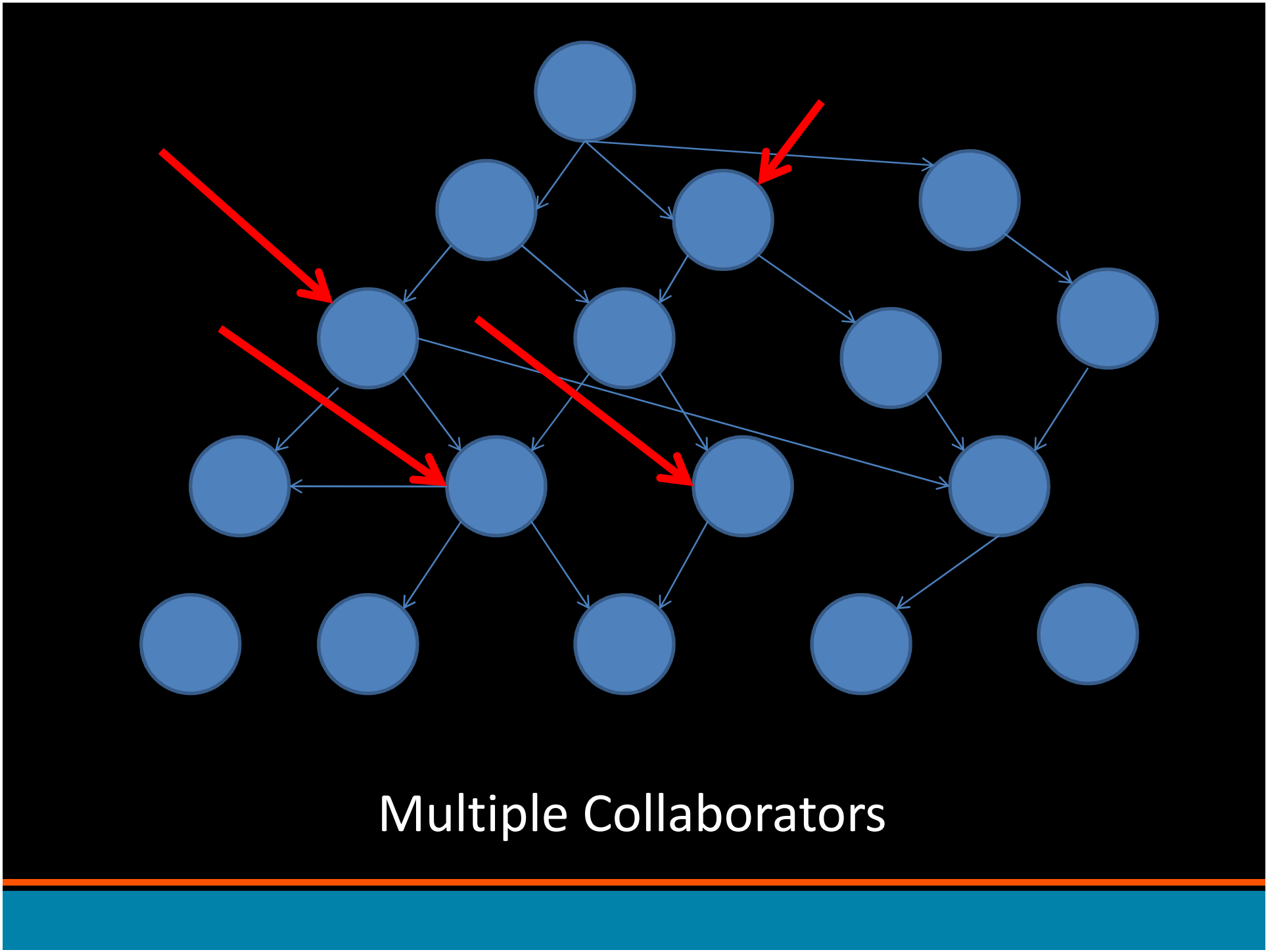
http://www.flickr.com/photos/mag3737/1914076277/



http://www.flickr.com/photos/gagilas/2659695352/

I’m against integration tests

Or at least 90% of them


Multiple Collaborators

Unmanageable nightmare

12 hour execution time?

5174 passed, 491 failed, 94 skipped

“Tests aren’t failing because of defects”

‘Nearly’ end-to-end

While knowing about everything in the middle...

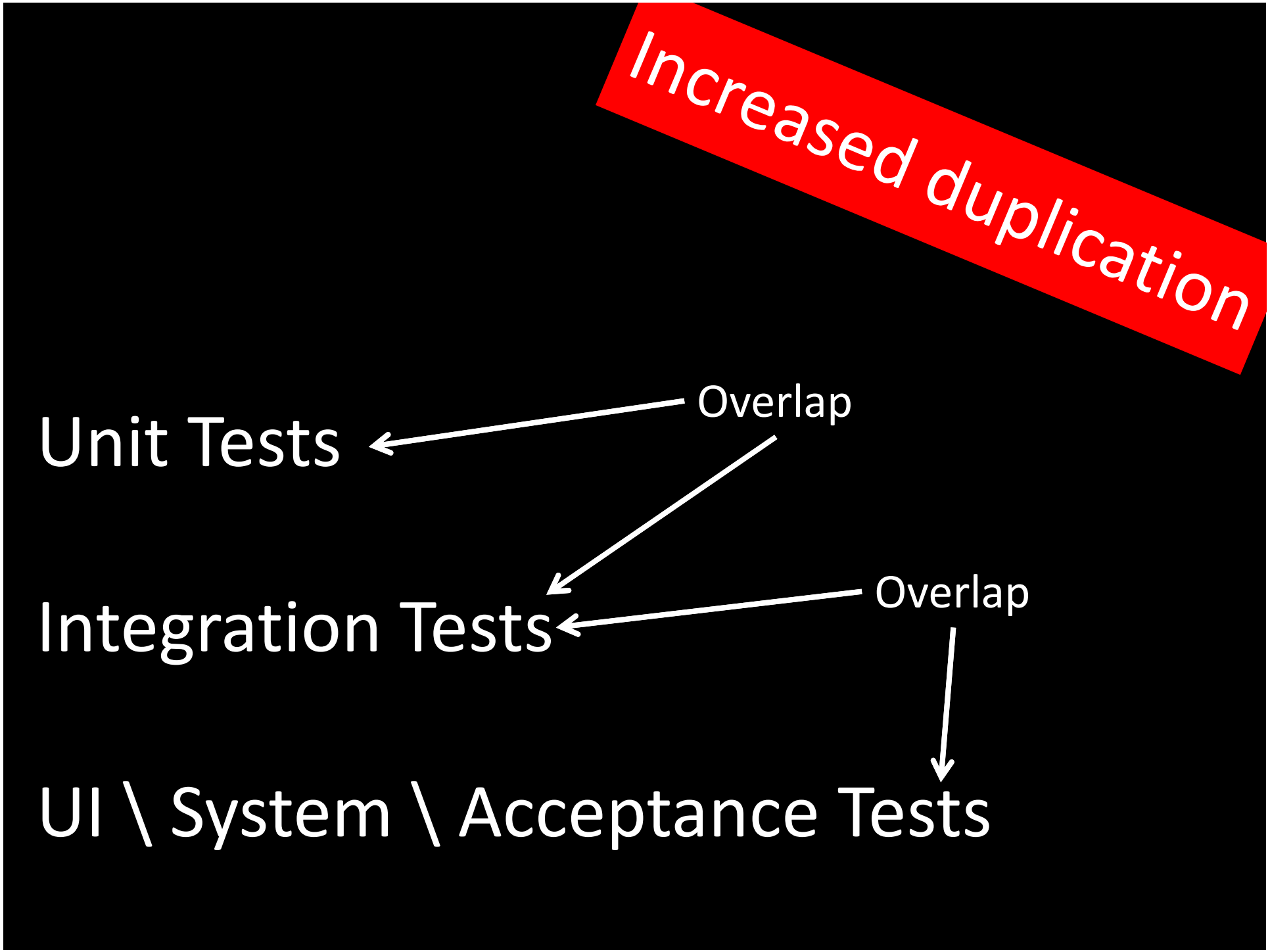
Becomes a pain to refactor

FEAR

FEAR

UNIT testing frameworks

Bad organisation

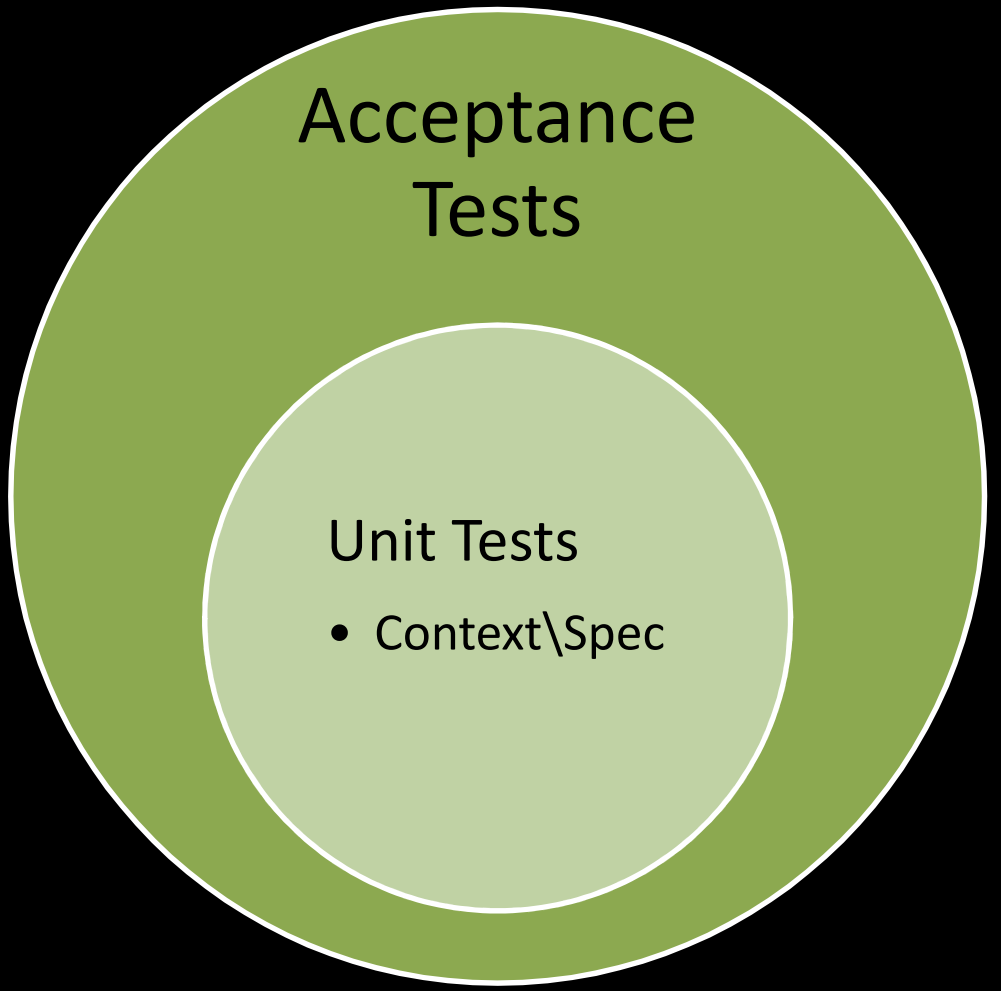

Increased duplication
Unit Tests
Overlap
Integration Tests
Overlap
UI \ System \ Acceptance Tests

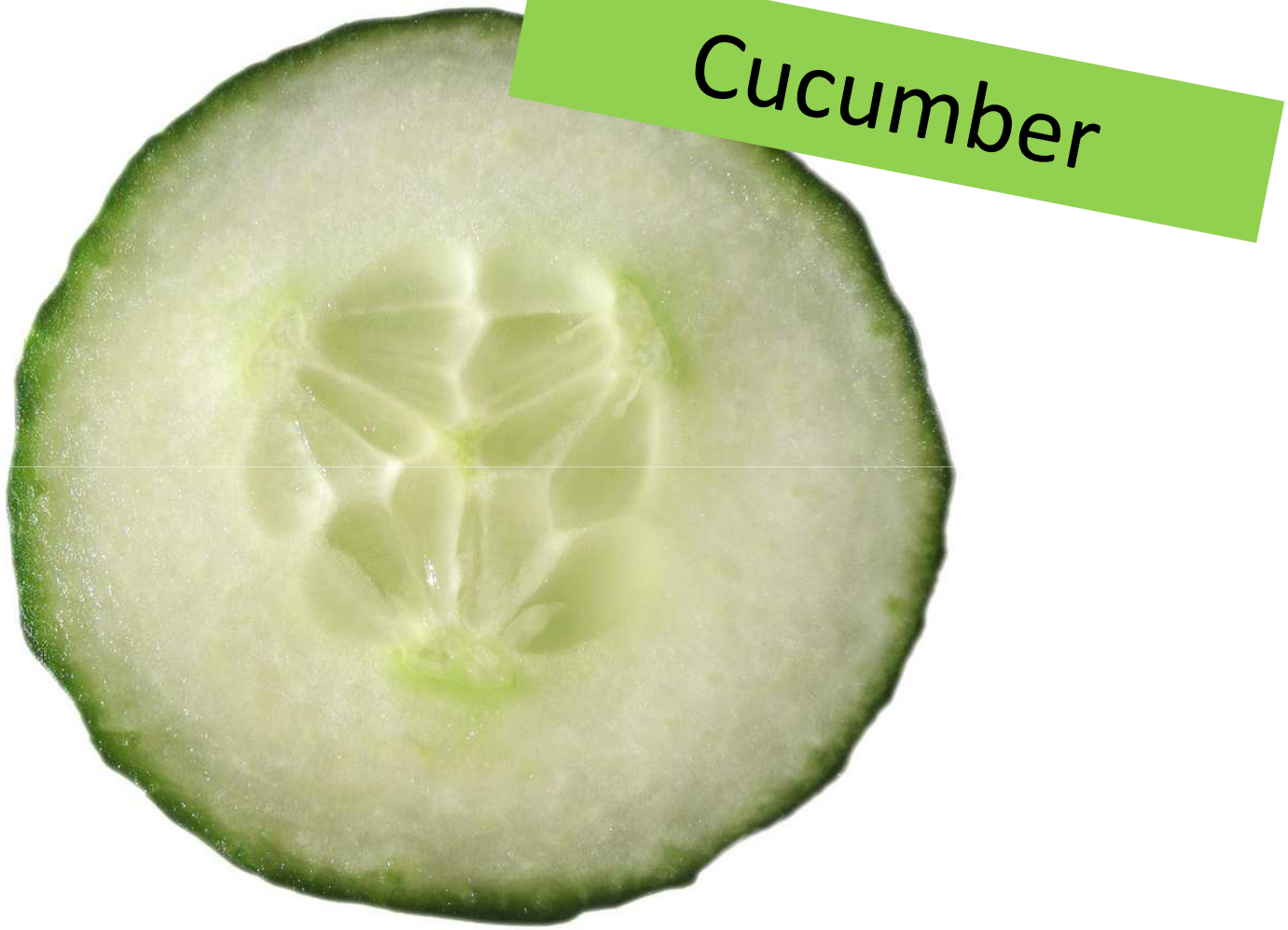
STOPPED

Unit Tests

Acceptance Tests

Outside-in Development





http://www.flickr.com/photos/vizzzual-dot-com/2738586453/

Feature: Creating a schedule

In Order to book meetings

The manager

Needs to be able to view a teams calendar

Scenario: View calendar of another view

Given “Bob” has a public calendar

When the manager views “Bob”’s calendar

Then an empty calendar should be shown

```
Given /^”([^\"]*)” has a public calendar$/ do |user|
  pending
End
When /^ the manager views “([^\"]*)”’s calendar$/
   do |user|
  pending
End
Then /^ an empty calendar should be shown $/ do
  pending
end
```

AdventureWorks The best gear on the net

Apparel | Hiking Equipment | Camping Gear | Climbing Equipment | Biking Gear | Featured Items

Favorites

- Lorem ipsum
- Aliquam diam
- Nam eu risus
- Proin arcu lectus

Recent

- Donec lacus
- Aenean id nunc
- Fusce non nibh
- Ut aliquet dictum
- Etiam eu est
- Aenean hendrerit
- Maecenas dapibus

Search Widget

Search

Featured

Adventure Works 20x30 Binoculars

Add to cart

Adventure Works 20x30 Binoculars

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aliquam nec nisi vel eros ornare auctor. Aenean vitae nulla. Sed in velit sit amet metus sollicitudin porttitor. Fusce non tortor. Nunc ornare imperdiet mauris. Nulla facilisi. In hac habitasse platea dictumst. Nullam ut odio eu nunc scelerisque tempor. Nulla facilisi. Quisque sem. Etiam risus dui, fringilla eget, imperdiet ac, vehicula in, nunc. Praesent pellentesque iaculis orci. Ut imperdiet dolor non turpis. In hac habitasse platea dictumst. Morbi neque sem, consequat sed, dignissim eget, ultrices aliquet, purus. Maecenas erat. Morbi commodo. Quisque nec purus. Duis dui.

Nunc non ante. Nullam ac diam. Nullam vel dolor eu ante pellentesque vulputate. Aliquam erat volutpat. Suspendisse a erat. In velit. Duis tellus lorem, porta eget, tempus a, volutpat vitae, nibh. Maecenas arcu dolor, adipiscing vel, tincidunt sit amet, mattis quis, metus. Vestibulum non justo. Ut ac nunc. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque dolor. Duis nec metus ut justo malesuada pellentesque. Etiam pretium leo ac leo. Curabitur nunc. Aliquam facilisis porta est. Cras hendrerit venenatis nisl. Sed placerat, urna sed condimentum mollis, ante elit sollicitudin est, at tempus odio dolor et quam.

Thanks to TekPub for Kona sample

AdventureWorks The best gear on the net

Apparel | Hiking Equipment | Camping Gear | Climbing Equipment | Biking Gear | Featured Items

Favorites

- Lorem ipsum
- Aliquam diam
- Nam eu risus
- Proin arcu lectus

Recent

- Donec lacus
- Aenean id nunc
- Fusce non nibh
- Ut aliquet dictum
- Etiam eu est
- Aenean hendrerit
- Maecenas dapibus

Search Widget

Search

Featured

Adventure Works 20x30 Binoculars

Add to cart

Adventure Works 20x30 Binoculars

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aliquam nec nisi vel eros ornare auctor. Aenean vitae nulla. Sed in velit sit amet metus sollicitudin porttitor. Fusce non tortor. Nunc ornare imperdiet mauris. Nulla facilisi. In hac habitasse platea dictumst. Nullam ut odio eu nunc scelerisque tempor. Nulla facilisi. Quisque sem. Etiam risus dui, fringilla eget, imperdiet ac, vehicula in, nunc. Praesent pellentesque iaculis orci. Ut imperdiet dolor non turpis. In hac habitasse platea dictumst. Morbi neque sem, consequat sed, dignissim eget, ultrices aliquet, purus. Maecenas erat. Morbi commodo. Quisque nec purus. Duis dui.

Nunc non ante. Nullam ac diam. Nullam vel dolor eu ante pellentesque vulputate. Aliquam erat volutpat. Suspendisse a erat. In velit. Duis tellus lorem, porta eget, tempus a, volutpat vitae, nibh. Maecenas arcu dolor, adipiscing vel, tincidunt sit amet, mattis quis, metus. Vestibulum non justo. Ut ac nunc. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque dolor. Duis nec metus ut justo malesuada pellentesque. Etiam pretium leo ac leo. Curabitur nunc. Aliquam facilisis porta est. Cras hendrerit venenatis nisl. Sed placerat, urna sed condimentum mollis, ante elit sollicitudin est, at tempus odio dolor et quam.

Thanks to TekPub for Kona sample

Feature: Display Products

In order to browse the catalog

The customer

Needs to see products

Scenario: Single featured product on homepage

Given the featured product "Adventure Works 20x30 Binoculars"

When I visit the homepage

Then I should see "Adventure Works 20x30 Binoculars" listed under "Featured"


WebRat
http://www.flickr.com/photos/whatwhat/22624256/

visit

click_link

fill_in

click_button

check and uncheck

choose

select

attach_file

```
Given /^the featured product "([^\"]*)"$/ do
  |product_name|
  Product.create product_name, ‘featured’
end
```

```
When /^I visit the homepage$/ do
  visit url + '/'
end
```

```
Then /^I should see "([^\"]*)" listed under "([^\"]*)"$/ do
    |product, area|
  within div_id(area) do |products|
    products.should contain(product)
  end
End


def div_id(area)
  "#" + area.downcase.underscore
end
```

env_selenium.rb

Dynamically replaced

```
def url
  "http://www.website.local"
end

Webrat.configure do |config|
  config.mode = :selenium
  config.application_framework = :external

  config.selenium_browser_key = get_browser_key
  puts "Executing tests using the browser #{config.selenium_browser_key}"
end

def get_browser_key()
  command = "*firefox"
  command = ENV['BROWSER'] if ENV['BROWSER']
  return command
end
```

```
cucumber --profile selenium browser=*firefox
cucumber --profile selenium browser=*safari
cucumber --profile selenium browser=*iexplore
cucumber --profile selenium browser=*googlechrome
```

HEADLESS BROWSING BY REPLACING ENV.RB

cucumber --profile webrat

Apparel | Hiking Equipment | Camping Gear | Climbing Equipment | Biking Gear | Featured Items

Favorites

- Lorem ipsum
- Aliquam diam
- Nam eu risus
- Proin arcu lectus

Recent

- Donec lacus
- Aenean id nunc
- Fusce non nibh
- Ut aliquet dictum
- Etiam eu est
- Aenean hendrerit
- Maecenas dapibus

North Star Compass

Price: £18.00

Add to cart

From The Manufacturer:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aliquam nec nisi vel eros ornare auctor. Aenean vitae nulla. Sed in velit sit amet metus sollicitudin porttitor. Fusce non tortor. Nunc ornare imperdiet mauris. Nulla facilisi. In hac habitasse platea dictumst. Nullam ut odio eu nunc scelerisque tempor. Nulla facilisi. Quisque sem. Etiam risus dui, fringilla eget, imperdiet ac, vehicula in, nunc. Praesent pellentesque iaculis orci. Ut imperdiet dolor non turpis. In hac habitasse platea dictumst. Morbi neque sem, consequat sed, dignissim eget, ultrices aliquet, purus. Maecenas erat. Morbi commodo. Quisque nec purus. Duis dui.

Nunc non ante. Nullam ac diam. Nullam vel dolor eu ante pellentesque vulputate. Aliquam erat volutpat. Suspendisse a erat. In velit. Duis tellus lorem, porta eget, tempus a, volutpat vitae, nibh. Maecenas arcu dolor, adipiscing vel, tincidunt sit amet, mattis quis, metus. Vestibulum non justo. Ut ac nunc. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque dolor. Duis nec metus ut justo malesuada pellentesque. Etiam pretium leo ac leo. Curabitur nunc. Aliquam facilisis porta est. Cras hendrerit venenatis nisl. Sed placerat, urna sed condimentum mollis, ante elit sollicitudin est, at tempus odio dolor et quam.

Technical Details:

Weight: XX
Material: XX
Height: XX
Width: XX
Weight: XX
Special Feature: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aliquam nec nisi vel eros ornare auctor. Aenean vitae nulla. Sed in velit sit amet metus sollicitudin porttitor. Fusce non tortor. Nunc ornare imperdiet mauris. Nulla facilisi. In hac habitasse platea dictumst. Nullam ut odio eu nunc scelerisque tempor. Nulla facilisi. Quisque sem. Etiam risus dui, fringilla eget, imperdiet ac, vehicula in, nunc. Praesent pellentesque iaculis orci. Ut imperdiet dolor non turpis. In hac habitasse platea dictumst. Morbi neque sem, consequat sed, dignissim eget

You might also like...

Thanks to TekPub for Kona sample

Feature: Recommending products

In order to increase conversions

The customer

Needs to be recommended to other products

Scenario: Recommendations should include other similar purchased products

Given customer order for "Trailhead Locking Carabiner" and "Climbing Rope with Single Caribiner"

And customer order for "Trailhead Locking Carabiner" and "North Star Compass"

When I visit the "Trailhead Locking Carabiner" details page

Then I should see "Climbing Rope with Single Caribiner" listed under "recommendations"

Then I should see "North Star Compass" listed under "recommendations"

```
Given /^customer order for "([^\"]*)"$/ do |product|
  order = Order.find_or_create
  OrderItem.create_item(order, product)
End

When /^I visit the "([^\"]*)" details page$/ do |product_name|
  product = Product.find_by_ProductName product_name
  visit "#{url}/Product/Details/#{product.SKU}"
end

Then /^I should see "([^\"]*)" listed$/ do |product|
  response.should contain(product)
end
```

```
Scenario Outline: Blowout Specials
  Given <product> in "Blowout Specials"
  When I visit the homepage
  Then I should see <product> listed under "Blowout Specials"

  Examples:
    | product                                  |
    | "Cascade Fur-lined Hiking Boots"         |
    | "Trailhead Locking Carabiner"            |
    | "Climbing Rope with Single Caribiner"    |
```

Multiple Hooks

```
Before do
  open_connection
  empty_database
end

at_exit do
  empty_database
end
```

Multiple extension frameworks

Test command line applications

Test Silverlight\Flex

Test WPF based desktop applications

Execute in parallel across multiple machines

Level of abstraction

HUGE RE-USE == VALUE

“There is no map”

Quote from Seth Godin - Linchpin

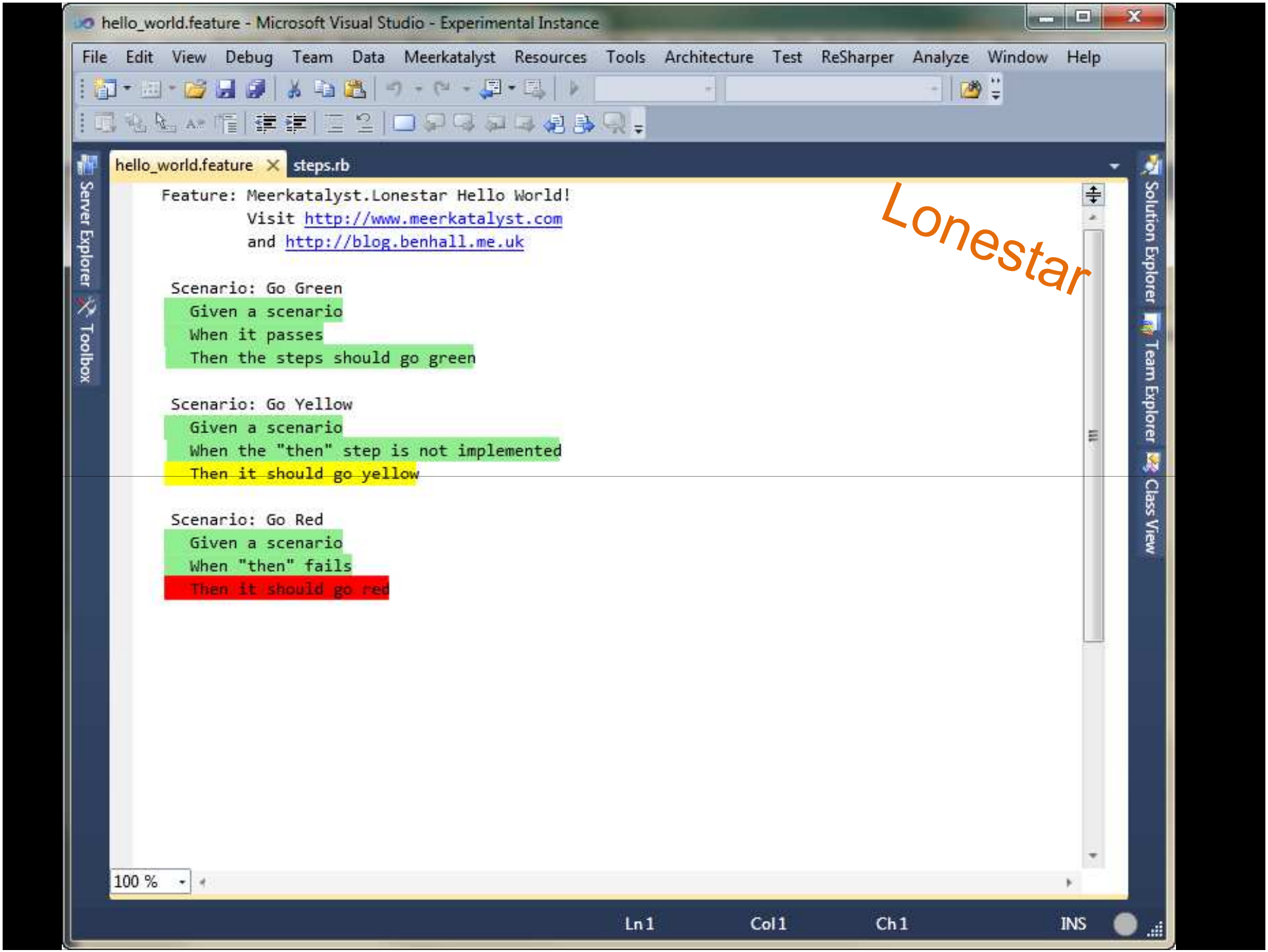
One more thing...


http://www.flickr.com/photos/leon_homan/2856628778/

Focus on finding true value

Look at other communities for advice, support and inspiration


@Ben_Hall
Ben@BenHall.me.uk
Blog.BenHall.me.uk
KTHXBAI
lolcatgenerator.com
http://lolcatgenerator.com/lolcat/120/


hello_world.feature - Microsoft Visual Studio - Experimental Instance
File Edit View Debug Team Data Meerkatalyst Resources Tools Architecture Test ReSharper Analyze Window Help
hello_world.feature
steps.rb
Feature: Meerkatalyst.Lonestar Hello World!
Visit http://www.meerkatalyst.com
and http://blog.benhall.me.uk
Scenario: Go Green
Given a scenario
When it passes
Then the steps should go green
Scenario: Go Yellow
Given a scenario
When the "then" step is not implemented
Then it should go yellow
Scenario: Go Red
Given a scenario
When "then" fails
Then it should go red
Lonestar
Server Explorer
Toolbox
Solution Explorer
Team Explorer
Class View
100 %
Ln 1
Col 1
Ch 1
INS

```
using Cuke4Nuke.Framework;
using NUnit.Framework;
using WatiN.Core;
namespace Google.StepDefinitions
{
    public class SearchSteps
    {
        Browser _browser;
        [Before]
        public void SetUp()
        {
            _browser = new WatiN.Core.IE();
        }
        [After]
        public void TearDown()
        {
            if (_browser != null)
            {
                _browser.Dispose();
            }
        }
        [When(@"^(?:I'm on|I go to) the search page$")]
        public void GoToSearchPage()
        {
            _browser.GoTo("http://www.google.com/");
        }
        [When("^I search for \"(.*)\"$")]
        public void SearchFor(string query)
        {
            _browser.TextField(Find.ByName("q")).TypeText(query);
            _browser.Button(Find.ByName("btnG")).Click();
        }
        [Then("^I should be on the search page$")]
        public void IsOnSearchPage()
        {
            Assert.That(_browser.Title == "Google");
        }
        [Then("^I should see \"(.*)\" in the results$")]
        public void ResultsContain(string expectedResult)
        {
            Assert.That(_browser.ContainsText(expectedResult));
        }
    }
}
```

```
Given /^(?:I'm on|I go to) the search page$/ do
  visit 'http://www.google.com'
end

When /^I search for "([^\"]*)"$/ do |query|
  fill_in 'q', :with => query
  click_button 'Google Search'
end

Then /^I should be on the search page$/ do
 dom.search('title').should == "Google"
end

Then /^I should see \"(.*)\" in the results$/ do |text|
  response.should contain(text)
end
```

Software

- Recommended:
  - IronRuby
  - Ruby
  - Cucumber
  - Rspec
  - WebRat
  - mechanize
  - Selenium RC
  - selenium-client
  - Caricature
  - activerecord-sqlserver-adapter
- Optional:
  - XSP \ Mono
  - JetBrain's RubyMine
  - JRuby
  - Capybara
  - Celerity
  - Active record
  - active-record-model-generator
  - Faker
  - Guid

Useful Links

- http://www.github.com/BenHall
- http://blog.benhall.me.uk
- http://stevehodgkiss.com/2009/11/14/using-activerecord-migrator-standalone-with-sqlite-and-sqlserver-on-windows.html
- http://www.testingaspnet.com
- http://
- http://msdn.microsoft.com/en-us/magazine/dd434651.aspx
- http://msdn.microsoft.com/en-us/magazine/dd453038.aspx
- http://www.cukes.info

SQL Server and Ruby

1. gem install activerecord-sqlserver-adapter
1. Download dbi-0.2.2.zip
2. Extract dbd\ADO.rb to ruby\site_ruby\1.8\DBD\ADO.rb