Anti-Patterns and Patterns in Rails


Chirantan Mitra a.k.a Chiku
Stares bad code into submission, every day.

Habibullah Pagarkar a.k.a **Habib**

In a rare good mood

@chirantanmitra

@mhabibp

Do you know this developer?

http://jeffmorin.blogspot.com/2011/02/sketch-of-week-when-it-all-goes-wrong.html

200 OK Please

Server responded => Success!

Very simple code

Inspect response body to get the error

flexible

```
$.ajax({
  url: this.href,
  dataType: 'json',
  type: 'POST',
  complete: widget.flexibleResponseHandler,
});

flexibleResponseHandler: function(data) {
  if (data.match('error')) {
    // blah blah
  }
  else {
    // some other blah blah
  }
}
```

Server side code

```
class FoosController < ApplicationController
  def create
    foo = Foo.new params[:foo]
    if foo.save
      render :json => foo
    else
      render :json => foo.errors
    end
  end
end
```

200 not OK

Rich list of HTTP status codes

Use the Framework, Luke!

422 is very under-rated

```
$.ajax({
  url: this.href,
  dataType: 'json',
  type: 'POST',
  complete: widget.flexibleResponseHandler,
});

flexibleResponseHandler: function(data) {
  if (data.status == 200) {
    // blah blah
  }
  else {
    // some other blah blah
  }
}
```

```
$.ajax({
  url: this.href,
  dataType: 'json',
  type: 'POST',
  success: function(response) {
    // blah blahh
  },
  error: function(response) {
    // some other blah blah
  },
  complete: function(response) {
    // yet more blah blah
  }
});
```

More responsible server side code

```
class FoosController < ApplicationController
  def create
    foo = Foo.new params[:foo]
    if foo.save
      render :json => foo
    else
      render :json => foo.errors,
:status => :unprocessable_entity
    end
  end
end
```

Segueing into a rant…*

Returning only 200 (OK) or

500 (Internal Server Error)

* Don’t mind us. We’ll be doing a few of these.

Inline JavaScript

I ❤ JavaScript with HTML

Readable and understandable with everything in one place

Who reads JavaScript?

And testing JavaScript? Puh-lease!

http://www.demotivationalposters.org/

Inline JavaScript

```
<a class="foo_description"
  href="#"
  onclick="new Ajax.Updater('foo_links',
      '/qux/9477/foo_links/36981?kind=Baz',{
        asynchronous:true,
        evalScripts:true
  });
  set_current_bar('Baz', 36981, 'WN2-1D');
  return false; "
>
WN2-1D - 296024
</a>
```

... Repeated 48 times ... (Sometimes 240 times)

Unobtrusive JavaScript

Decouple behaviour from HTML

Testable

Rails 3 has moved away from it

Reduce payload

http://www.flickr.com/photos/g-mikee/4400087493/

```
<a class="foo_description" href="#" data-qux-id="9477" data-
id="36981" data-magic="WN2-1D"> WN2-1D - 296024 </a>
```

```
$$(".foo_description").each(function(link) {
  var id = link.readAttribute('data-id');
  var url = '/qux/' + link.readAttribute('data-qux-id') + '/
foo_links/' + id + '?kind=Baz';

  link.observe("click", function() {
    new Ajax.Updater('foo_links', url,
      { asynchronous:true, evalScripts:true });
    set_current_bar('Baz', id, link.readAttribute('data-
magic'));
    return false;
    });
});
```

```
function Foo(targetSelector) {
  this.target = $$(targetSelector);
  this.initialize();
}

Foo.prototype = {
  initialize: function() {
    this.quxes = this.target.map(function(link) {
      return new Qux(link);
    });
  }
}

function Qux(target) {
  this.target = target;
  this.initialize();
}
```

```
Qux.prototype = {
  initialize: function() {
    this.bind();
    this.setCurrentBar();
    return false;
  },
  bind: function() {
    var that = this;
    target.observe("click", function() {
      new Ajax.Updater('foo_links',that.url,
{evalScripts:true});
      this.setCurrentBar();
      return false;
    });
  },
  setCurrentBar: function() {
set_current_bar('Baz',this.id(), this.target.readAttribute
('data-magic'));},
  id: function() {return this.target.readAttribute('data-
id');},
  url: function() {
return ('/qux/' + this.target.readAttribute('data-qux-id') +
'/foo_links/' + this.id() + '?kind=Baz');}}
```

Validation in Controllers

Fat models aren't pretty

Controller checks if input is valid, that's all.
What's the harm in that?

This makes my tests very thorough

I can give custom error messages

```ruby
class ConflictCreationsController < ApplicationController
  def create_conflict
    render_unprocessable_entity("Are") and return if params[:op].blank?

    if params[:dn].blank? and params[:ocn].blank?
      render_unprocessable_entity("You") and return
    end

    # ... more of these ...

    which_1 = params[:dn] || params[:ocn]
    got_ya = CrazyStuff.find_by_which_1(which_1)

    render_unprocessable_entity("Really") and return unless got_ya

    got_ya.conflict

    if got_ya.conflicted?
      render :xml => got_ya, :status => :created
    else
      render_unprocessable_entity(got_ya.errors)
    end
  end
end
```

No Validation in Controllers

Whoa! Fat Models >> Fat Controllers

Validations should sit within a model

Controller tests should be simple

```ruby
class ConflictCreation
  include ActiveModel::Validations

  attr_accessor :op, :dn, :ocn, :got_ya

  validates_presence_of :op, :got_ya
  validate :dn_or_ocn_is_present

  def initialize(options = {})
    @op  = options[:op]
    @dn  = options[:dn]
    @ocn = options[:ocn]
    assign_got_ya
    conflict
  end
end
```

```ruby
class ConflictCreation
  def assign_got_ya
    which_1 = dn || ocn
    @got_ya = CrazyStuff.find_by_which_1(which_1)
  end

  def conflict
    got_ya and got_ya.conflict_with op
  end

  def dn_or_ocn_is_present
    if dn.blank? and ocn.blank?
      errors[:base].push "You"
    end
  end
end
```

```ruby
class ConflictCreationsController <
          ApplicationController
  def create
    conflict_creation=ConflictCreation.new params

    if conflict_creation.valid?
      render :xml => conflict_creation.got_ya,
             :status => :created
    else
      render_unprocessable_entity(
conflict_creation.errors)
    end
  end
end
```

Fact: Database calls are slow

Fast feedback required for tests

Solution: Mock all DB calls!

Mocks over mocks to go faster

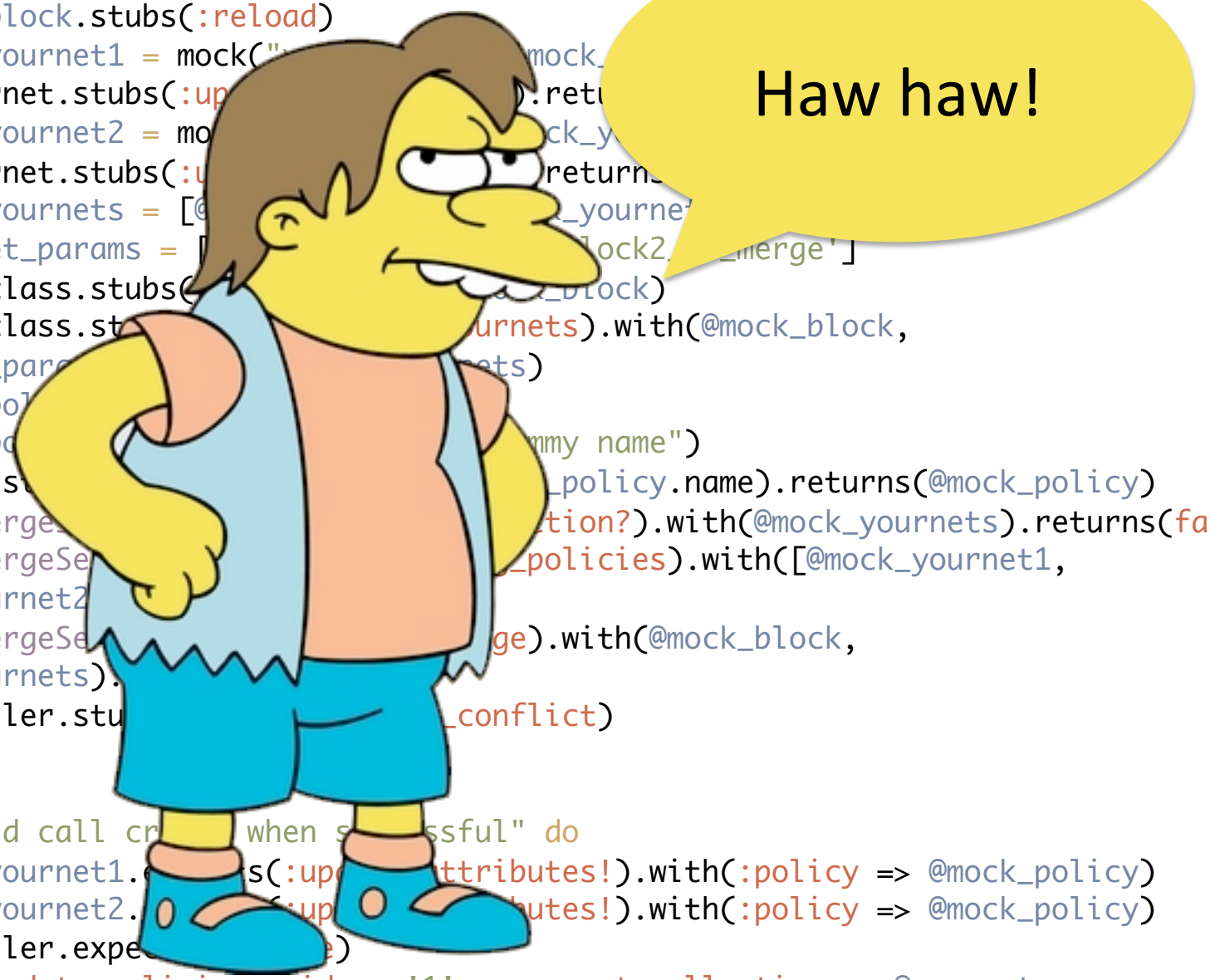
Stub everything, just in case

```
before(:each) do
  @mock_block = mock("block")
  @mock_block.stubs(:reload)
  @mock_yournet1 = mock("yournet1") { |mock_yournet|
mock_yournet.stubs(:update_attributes!).returns }
  @mock_yournet2 = mock("yournet2") {|mock_yournet|
mock_yournet.stubs(:update_attributes!).returns }
  @mock_yournets = [@mock_yournet1, @mock_yournet2]
  @yournet_params = ['block1_to_merge', 'block2_to_merge']
  block_class.stubs(:find).returns(@mock_block)
  block_class.stubs(:get_selected_yournets).with(@mock_block,
@yournet_params).returns(@mock_yournets)
  @mock_policy = mock("policy")
  @mock_policy.stubs(:name).returns("dummy name")
  Policy.stubs(:find_by_name).with(@mock_policy.name).returns(@mock_policy)
  BlockMergeService.stubs(:invalid_selection?).with(@mock_yournets).returns(false)
  BlockMergeService.stubs(:conflicting_policies).with([@mock_yournet1,
@mock_yournet2]).returns([])
  BlockMergeService.stubs(:try_to_merge).with(@mock_block,
@mock_yournets).returns(true)
  controller.stubs(:process_policy_conflict)
end

it "should call create when successful" do
  @mock_yournet1.expects(:update_attributes!).with(:policy => @mock_policy)
  @mock_yournet2.expects(:update_attributes!).with(:policy => @mock_policy)
  controller.expects(:create)
  post :update_policies, :id => '1', :yournet_collection => @yournet_params,
:selected_policy => @mock_policy.name
end
```

```
before(:each) do
  @mock_block = mock("block")
  @mock_block.stubs(:reload)
  @mock_yournet1 = mock("
mock_yournet.stubs(:up
  @mock_yournet2 = mo
mock_yournet.stubs(:u
  @mock_yournets = [@
  @yournet_params = [
  block_class.stubs(
  block_class.st
@yournet_para
  @mock_pol
  @mock_po
  Policy.s
  BlockMerge
  BlockMergeSe
@mock_yournet2
  BlockMergeSe
@mock_yournets).
  controller.stu
end

it "should call cr
  @mock_yournet1.
  @mock_yournet2.
  controller.expe
  post :update_policies, :id => '1', :yournet_collection => @yournet_params,
:selected_policy => @mock_policy.name
end
```



http://en.wikipedia.org/wiki/File:Nelson_Muntz.PNG

Now you are just testing your mocks

Test behavior and not implementation

Makes refactoring painful

Mock and stub with care

IN AMERICA, YOU MOCK CODE.

IN SOVIET RUSSIA,
CODE MOCK YOU!!

method_missing

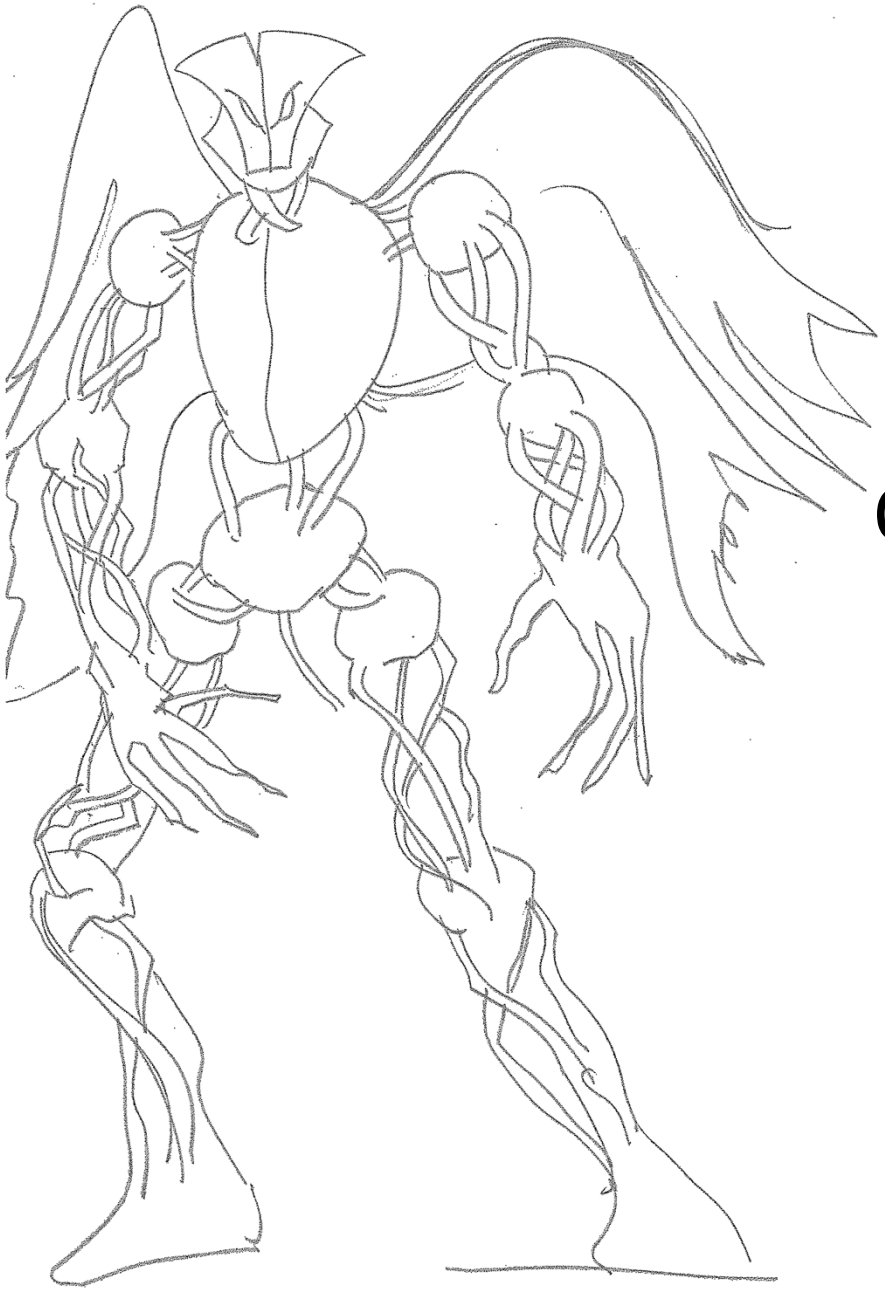
method_missing is the best delegator a programmer can have

Leads to DRY code

It is *elegant*

```ruby
module MethodMissing
  class << self
    def config_hash(hash={'foo' => 'foo', 'bar' => 'bar'})
      @configuration = hash
    end

    def method_missing(method_id)
      config_option = method_id.id2name.to_s
      raise "Booga (#{config_option}) or Ooga." unless ['foo',
'bar', 'baz', 'qux', 'quux', 'corge'].include?(config_option)
      @configuration[config_option]
    end
  end
end
```



method_found

Maintenance nightmare

Often badly implemented

Ugly stack-trace

Slow dispatch

```ruby
module MethodMissing
  class << self
    def config_hash(hash={'foo' => 'foo', 'bar' => 'bar'})
      @configuration = hash
    end

    def method_missing(method_id, &block)
      config_option = method_id.to_s
      if self.argument_valid?(config_option)
        @configuration[config_option]
      else
        super
      end
    end

    def respond_to?(method_id)
      config_option = method_id.to_s
      argument_valid?(config_option) || super
    end

    def argument_valid?(argument)
      ['foo', 'bar', 'baz', 'qux', 'quux', 'corge'].include?(argument)
    end
  end
end
```

```ruby
module MethodMissing
  class << self
    def config_hash(hash={'foo' => 'foo', 'bar' =>
'bar'})
      @configuration = hash
    end

    def method_missing(method_id, &block)
      config_option = method_id.to_s
      if self.argument_valid?(config_option)
        @configuration[config_option]
      else
        super
      end
    end
  end
end
```

```ruby
module MethodMissing
  class << self
    def respond_to?(method_id)
      config_option = method_id.to_s
      argument_valid?(config_option) || super
    end

    def argument_valid?(argument)
      ['foo', 'bar', 'baz', 'qux', 'quux',
'corge'].include?(argument)
    end
  end
end
```

Without method_missing

```
require 'forwardable'
require 'ostruct'

module NoMethodMissing
  class << self

    extend Forwardable
    def_delegators :@configuration, :foo, :bar

    def config_hash(hash={'foo' => 'foo', 'bar' =>
'bar'})
      @configuration = OpenStruct.new hash
    end
  end
end
```

Simplicity

```
def foo
  @configuration['foo']
end

def bar
  @configuration['bar']
end
```

Meta-programming light-saber

Ruby is all about meta-programming

It leads to DRY[#] code

How will people know how **cool** I am?

[#] dry_count.succ

You don't know the power of the dark side!

http://en.wikipedia.org/wiki/Darth_Vader

```ruby
def validate_x_must_have_y_if_foo
  validate_x_has_y_if_not_foo(:number)
end

def validate_x_has_y_if_not_foo(
  attr,
  error_message="\"of '\#{self}' (\#{self.send(attr)})
blah sn '\#{sn}' (\#{sn.send(attr)}) if sn is not
subnd\"")

  return unless errors.empty?

  return if subnd?

  errors.add(
    attr,
    instance_eval(error_message)
  ) if sn && !sn.send (attr).blank? && sn.send(attr)
!= self.send(attr)
end
```

Episode I: The Eval Menace

```
Address.module_eval do
  State.constants.each do |state_var|
    eval(<<-SRC
      def #{state_var.downcase}?
        state.self_and_ancestors.any? do |state|
          state == State::#{state_var}
        end
      end
    SRC
 )
  end
end
```

Episode II: Attack of the Evals

```
<%=
  form.select
    'from_company_id',
    ApplicationHelper.with_prompt(from_companies),
    {},
    { :onchange => eval(@order.store.r_f) }
%>
```

Episode III: Revenge of the Eval

```
selected_customer_ids =
  eval(params[:customer][:selected] ||
"[]")
```

:acts_as_light_saber

Should reduce complexity –
not increase it

When in doubt…ask. Please!

Use in moderation

http://en.wikipedia.org/wiki/Obi-Wan_Kenobi

```ruby
def validate_x_must_have_y_if_foo
  return false if errors.present?

  return true if subnd?

  if sn && !sn.send(attr).blank? &&
      sn.send(attr) != self.send(attr)
      message = <<-EOS
  of '#{self}' (#{self.send(attr)}) must be identical to
that of sn '#{sn}' (#{sn.send(attr)}) if sn is not subnd
EOS
      errors.add(:foo, message) and false
  end
end
```

Episode IV: A New Hope

```
 Address.module_eval do
    State.constants.each do |state_var|
      define_method "#{state_var.downcase}?" do
        state.self_and_ancestors.any? do |state|
state == eval("State::#{state_var}")
        end
      end
    end
  end
```

Episode VI: Return of the ..umm..

```
Address.module_eval do
    State.constants.each do |state_var|
      define_method "#{state_var.downcase}?" do
        state.self_and_ancestors.any? do |state|
state == "Address::State::#
{state_var}".constantize
        end
      end
    end
  end
```

Copy code from TextMate with syntax highlighting

https://github.com/drnic/copy-as-rtf-tmbundle

All the cool angel and devil artwork by

Avishek ‘Mojo’ Sen Gupta

http://www.avishek.net

@chirantanmitra

@mhabibp

Thank You!

Questions?