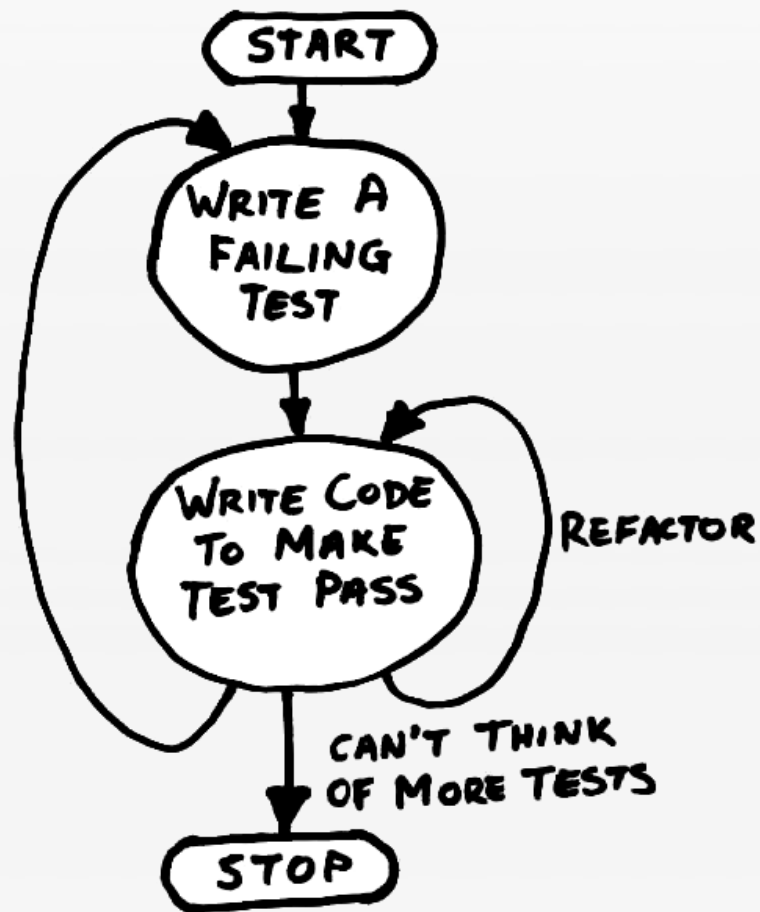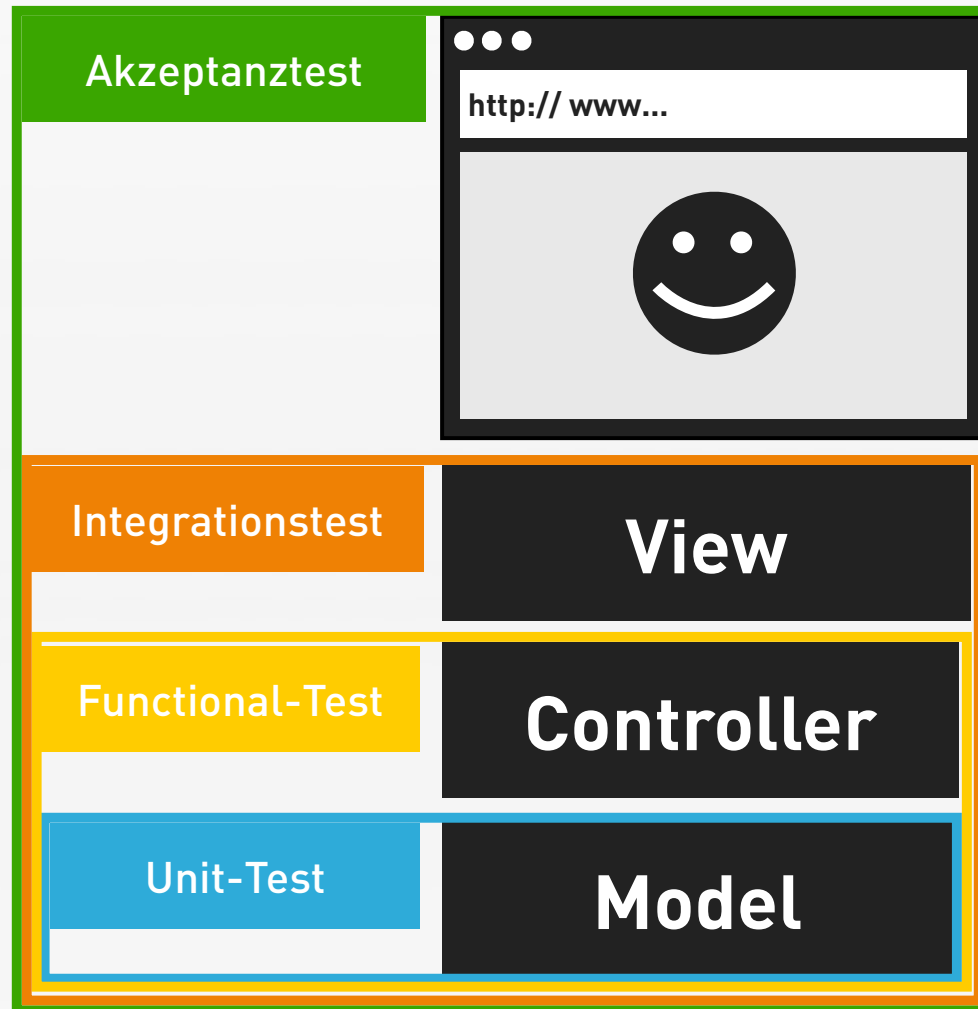# Vorlesung 8

Warum TDD?

- Wasserfallmodell: Testphase erst nach Umsetzungsphase

- Probleme:

  - Testabdeckung oft mangelhaft

  - Entwickelte Komponenten sind schwierig zu testen

  - Entwicklung neuer Features führt oft zu Regressionsfehlern

- Testgetriebene Entwicklung: Gemeinsame Entwicklung von Tests und Features

TDD-Zyklus

Test einer RoR-Anwendung

RECAP

RSpec - Expectations

```ruby
require 'rspec'

class Calculator
  def plus(n1, n2)
    n1 + n2
  end
end

describe Calculator do
  it 'calculates sums' do
    calc = Calculator.new
 !  calc.plus(19, 23).should == 42
  end
end
```

RSpec - Mocks

```ruby
require 'rspec'

class Calculator
  def plus(n1, n2)
    n1 + n2
  end
end


describe Calculator do
  it 'calculates sums' do
    calc = Calculator.new
    calc.stub(:plus).and_return(42)
    calc.plus(19, 23).should == 42
  end
end
```

User Story 2

- Test anlegen

```
                              spec/models/tour_spec.rb

require 'rspec'

describe Tour do
  it 'needs a category' do
    category = Category.new name: 'Radwandern'
    tour = Tour.new title: 'Test Tour', teaser: 'My Teaser', description: 'My
description', url: 'http://www.google.com/maps/'

    tour.should_not be_valid

    tour.categories << category
    tour.should be_valid
  end
end
```

User Story 2

- Test ausführen

User Story 2

- Code schreiben

app/models/tour.rb

```ruby
class Tour < ActiveRecord::Base
  has_many :tour_categories
  has_many :categories, through: :tour_categories
  validates :title, :teaser, :description, :url, presence: true
  validates :url, format: %r|^http(s?)://www.google.com/maps/ |
! validates :categories, presence: true
end
```

User Story 2

- Test ausführen

User Story 2

- ● Test refactorn

spec/models/tour_spec.rb

```ruby
require 'rspec'

describe Tour do
  let!(:category) { Category.new name: 'Radwandern' }
  let!(:tour) { Tour.new title: 'Test Tour', teaser: 'My Teaser', description: 'My description',
url: 'http://www.google.com/maps/' }

  it 'is not valid without a category' do
    tour.should_not be_valid
  end

  it 'is valid with a category' do
    tour.categories << category
    tour.should be_valid
  end
end
```

Taskboard

Inbox	In Progress	In Review	Complete
Ein *Anwendung* soll über ein einfaches Styling verfügen	Ein *Besucher* soll *Touren nach Kategorie filtern* können, ...	Ein *Besucher* soll *Touren anlegen* können, ...	

User Story 2
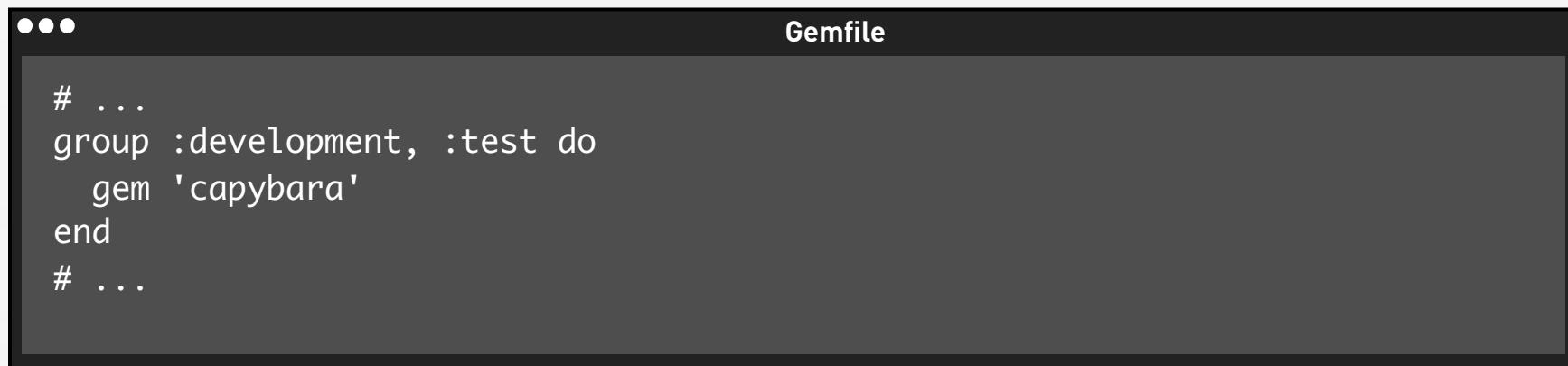
Ein *Besucher* soll *Touren nach Kategorie filtern* können, um schneller die passende Tour zu finden.

Integration-Test:
Ein Besucher soll alle Touren auf der Startseite sehen

Voraussetzungen: Capybara

- Gem 'capybara' ins Gemfile aufnehmen:

```
                                    Gemfile

# ...
group :development, :test do
  gem 'capybara'
end
# ...
```

```
$ bundle install
```

Capybara

- Capybara bietet eine domänenspezifische Sprache (DSL) zur Steuerung eines virtuellen Browsers

- RSpec startet zum Testen einen Testserver (~ rails server)

- Capybara greift auf diesen Testserver zu

Capybara

- **Steuerbefehle:**

  - Seite '/tours' aufrufen:
    ```
    visit '/tours'
    ```

  - Link klicken:
    ```
    click_link 'New tour'
    ```

  - Formular ausfüllen und abschicken:
    ```
    fill_in 'Name', with: 'Münsterland Giro'
    click_button 'Create tour'
    ```

- **Expectations:**

  - Seite soll bestimmten Inhalt haben:
    ```
    page.should have_content 'Created!'
    ```

1. Test schreiben

```ruby
                                    spec/features/tours_spec.rb

require 'spec_helper'

describe "Tours" do
  context 'start page' do
    let!(:category) { Category.new name: 'Category' }
    let!(:tour) { Tour.new title: 'Test Tour', teaser: 'My Teaser',
description: 'My description', url: 'http://www.google.com/maps/' }

    before(:each) do
      tour.categories << category
      tour.save
    end

    it 'shows all tours on the start page' do
      visit root_path

      page.should have_content tour.title
    end
  end
end
```

2. Tests ausführen

```
thomas@t420: ~/Coding/Ruby/we2/leeze.ms

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rspec spec
..*.......*F

Pending:
  TourCategory add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/tour_category_spec.rb
    # No reason given
    # ./spec/models/tour_category_spec.rb:4
  Category add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/category_spec.rb
    # No reason given
    # ./spec/models/category_spec.rb:4

Failures:

  1) Tours start page shows all tours on the start page
     Failure/Error: visit root_path
     NameError:
       undefined local variable or method `root_path' for #<RSpec::Core::ExampleGroup::Nested_4::Nested_1:0x0000000353ff18>
     # ./spec/requests/tours_spec.rb:14:in `block (3 levels) in <top (required)>'

Finished in 0.30006 seconds
12 examples, 1 failure, 2 pending

Failed examples:

rspec ./spec/requests/tours_spec.rb:13 # Tours start page shows all tours on the start page

Randomized with seed 10207

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] %                                    :(
```

3. Code schreiben

- public/index.html löschen

- Root-Route eintragen:

```
config/routes.rb

LeezeMs::Application.routes.draw do
  resources :tours
  root 'tours#index'
end
```

4. Test ausführen

5. Refactoring

- Wir haben zwei Tests, in denen eine Tour und eine Kategorie erstellt werden
  --> nicht DRY!

- Auch Tests sollten refactored werden!

- Einführung von Factories

factory_girl

- factory_girl erlaubt das Erstellen von Factories (Fabriken) für Models

- Eine Factory erzeugt eine Instanz des Models mit sinnvollen Standardwerten

- Factories werden in spec/factories/ erstellt

factory_girl

Factory definieren:

```
FactoryGirl.define do
  factory :category do
    name "My Category"
  end
end
```

Factory nutzen:

```
FactoryGirl.build(:category)
# ~> Category.new name: "My Category"

FactoryGirl.create(:category)
# ~> Category.create name: "My Category"

FactoryGirl.create(:category, name: "Some other name")
# ~> Category.new name: "Some other name"
```

5. Refactoring

Factories definieren:

spec/factories/category.rb

```ruby
FactoryGirl.define do
  factory :category do
    sequence(:name) {|n| "My category #{n}" }
  end
end
```

5. Refactoring

Factories definieren:

```
                           spec/factories/tour.rb

FactoryGirl.define do
  factory :tour do
    title "My Tour"
    teaser "My Teaser"
    description "My Description"
    url "http://www.google.com/maps/"
    categories { [FactoryGirl.build(:category)] }
  end
end
```

5. Refactoring

Factories einbauen:

```ruby
                                    spec/models/tour_spec.rb

require 'spec_helper'

describe Tour do
  let!(:category) { FactoryGirl.build(:category) }
  let!(:tour) { FactoryGirl.build(:tour, categories: []) }

  it 'is not valid without a category' do
    tour.should_not be_valid
  end


  it 'is valid with a category' do
    tour.categories << category
    tour.should be_valid
  end
end
```

5. Refactoring

Factories einbauen:

```ruby
require 'spec_helper'

describe "Tours" do
  context 'start page' do
    let!(:tour) { FactoryGirl.create(:tour) }

    it 'shows all tours on the start page' do
      visit root_path

      page.should have_content tour.title
    end
  end
end
```

5. Refactoring

Tests ausführen:

```
thomas@t420: ~/Coding/Ruby/we2/leeze.ms

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rspec spec
.*.........*

Pending:
  Category add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/category_spec.rb
    # No reason given
    # ./spec/models/category_spec.rb:4
  TourCategory add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/tour_category_spec.rb
    # No reason given
    # ./spec/models/tour_category_spec.rb:4

Finished in 0.49769 seconds
12 examples, 0 failures, 2 pending

Randomized with seed 51457

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] %
```

User Story 2

Ein *Besucher* soll *Touren nach Kategorie filtern* können, um schneller die passende Tour zu finden.

1. Test schreiben

```ruby
require 'spec_helper'

describe "Tours" do
  context 'start page' do
    # ...
  end


  context 'filter' do
    let!(:tour1) { FactoryGirl.create(:tour, title: 'Tour 1') }
    let!(:category1) { tour1.categories.first }
    let!(:tour2) { FactoryGirl.create(:tour, title: 'Tour 2') }
    let!(:category2) { tour2.categories.first }

    it 'allows to select categories' do
      visit root_path
      click_link category1.name
      page.should have_content tour1.title
      page.should_not have_content tour2.title
    end
  end
end
```

2. Tests ausführen

```
○ ○ ○   thomas@t420: ~/Coding/Ruby/we2/leeze.ms
thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rspec spec
..**........F

Pending:
  Category add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/category_spec.rb
    # No reason given
    # ./spec/models/category_spec.rb:4
  TourCategory add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/tour_category_spec.rb
    # No reason given
    # ./spec/models/tour_category_spec.rb:4

Failures:

  1) Tours filter allows to select categories
     Failure/Error: click_link category1.name
     Capybara::ElementNotFound:
       no link with title, id or text 'My Category' found
     # (eval):2:in `click_link'
     # ./spec/requests/tours_spec.rb:24:in `block (3 levels) in <top (required)>'

Finished in 0.54781 seconds
13 examples, 1 failure, 2 pending

Failed examples:

rspec ./spec/requests/tours_spec.rb:21 # Tours filter allows to select categories

Randomized with seed 40529

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % ▮                                          :(
```
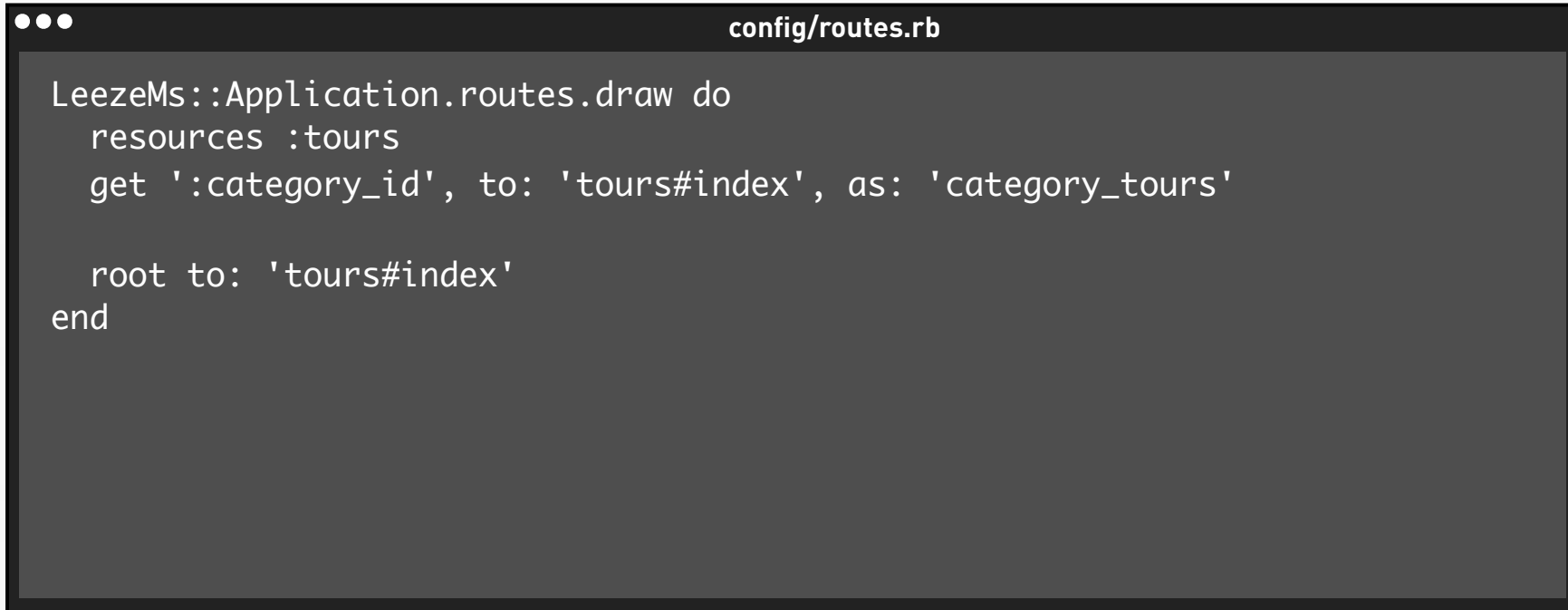
3. Code schreiben

config/routes.rb

```ruby
LeezeMs::Application.routes.draw do
  resources :tours
  get ':category_id', to: 'tours#index', as: 'category_tours'

  root to: 'tours#index'
end
```

3. Code schreiben

```
●●●  thomas@t420: ~/Coding/Ruby/we2/leeze.ms
thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rake routes
         tours GET     /tours(.:format)              tours#index
               POST    /tours(.:format)              tours#create
      new_tour GET     /tours/new(.:format)          tours#new
     edit_tour GET     /tours/:id/edit(.:format)     tours#edit
          tour GET     /tours/:id(.:format)          tours#show
               PUT     /tours/:id(.:format)          tours#update
               DELETE  /tours/:id(.:format)          tours#destroy
category_tours GET     /:category_id(.:format)       tours#index
          root /                                     tours#index
thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % █
```

3. Code schreiben

app/controllers/tours_controller.rb

```ruby
class ToursController < ApplicationController
  # GET /tours
  # GET /tours.json
  def index
    @tours = if params[:category_id]
               Category.find(params[:category_id]).tours
             else
               Tour.all
             end


    respond_to do |format|
      format.html # index.html.erb
      format.json { render json: @tours }
    end
  end


  # ...
end
```

3. Code schreiben

```erb
<h1>Listing tours</h1>

Categories:
<ul>
  <% Category.all.each do |category| %>
    <li>
      <%= link_to category.name, category_tours_path(category) %>
      </li>
  <% end %>
</ul>

...
```

4. Tests ausführen


```
thomas@t420: ~/Coding/Ruby/we2/leeze.ms

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rspec spec
.......*....*

Pending:
  TourCategory add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/tour_category_spec.rb
    # No reason given
    # ./spec/models/tour_category_spec.rb:4
  Category add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/category_spec.rb
    # No reason given
    # ./spec/models/category_spec.rb:4

Finished in 0.59092 seconds
13 examples, 0 failures, 2 pending

Randomized with seed 40362

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] %
```

5. Refactoring

- Problem: Die generierten URLs sind hässlich

  - http://localhost:3000/1

- Lösung: Name der Kategorie an die ID anhängen

  - http://localhost:3000/1-radtour

5. Refactoring

```
app/models/category.rb

class Category < ActiveRecord::Base
  attr_accessible :name
    has_many :tour_categories
    has_many :tours, through: :tour_categories

  def to_param
    "#{id}-#{name.parameterize}"
  end
end
```

5. Refactoring

- Tests ausführen:

```
thomas@t420: ~/Coding/Ruby/we2/leeze.ms

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % rspec spec
........*....*

Pending:
  TourCategory add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/tour_category_spec.rb
    # No reason given
    # ./spec/models/tour_category_spec.rb:4
  Category add some examples to (or delete) /home/thomas/Coding/Ruby/we2/leeze.ms/spec/models/category_spec.rb
    # No reason given
    # ./spec/models/category_spec.rb:4

Finished in 0.59092 seconds
13 examples, 0 failures, 2 pending

Randomized with seed 40362

thomas@t420 ~/Coding/Ruby/we2/leeze.ms (git)-[2_filter_by_categories] % █
```

Taskboard

Inbox	In Progress	In Review	Complete
Ein *Anwendung* soll über ein einfaches Styling verfügen		Ein *Besucher* soll *Touren anlegen* können, …	
	Ein *Besucher* soll *Touren nach Kategorie filtern* können, …	Ein *Besucher* soll *Touren nach Kategorie filtern* können, …	

Taskboard

Inbox	In Progress	In Review	Complete
Ein Anwendung soll über ein einfaches Styling verfügen	Ein Anwendung soll über ein einfaches Styling verfügen	Ein Besucher soll Touren anlegen können, ... Ein Besucher soll Touren nach Kategorie filtern können, ...	

User Story 3

Die Anwendung soll über ein einfaches Styling verfügen

Twitter Bootstrap

Benötigte Gems einbauen:

```
                                          Gemfile
# ...
group :asset do
  gem 'therubyracer'
  gem 'twitter-bootstrap-rails'
  gem 'less-rails'
end
# ...
```

```
$ bundle install
```

Twitter Bootstrap

- **Nötige Assets in die Anwendung kopieren:**
  ```
  $ rails generate bootstrap:install
  ```

- **Application Layout durch Bootstrap ersetzen:**
  ```
  $ rails generate bootstrap:layout
  application fluid
  ```

Twitter Bootstrap

Tabelle auf der Übersichtsseite anzeigen:

```
app/views/tours/index.html.erb

<table>
  <tr>
    <th>Title</th>
    <th></th>
    <th></th>
    <th></th>
  </tr>
<% @tours.each do |tour| %>
  <tr>
    <td><%= tour.title %></td>
    <td><%= link_to 'Show', tour %></td>
    <td><%= link_to 'Edit', edit_tour_path(tour) %></td>
    <td><%= link_to 'Destroy', tour, method: :delete, data: { confirm: 'Are you
sure?' } %></td>
  </tr>
<% end %>
</table>


<br />
<%= link_to 'New Tour', new_tour_path %>
```

Twitter Bootstrap