Introduction to Ruby on Rails

Software Engineering II
WS 2018/19

Ralf Teusner
ralf.teusner@hpi.de

Prof. Plattner, Dr. Uflacker
Enterprise Platform and Integration Concepts group
Introduction to Ruby on Rails

1. Ruby & Ruby on Rails
   ■ What is Ruby on Rails?
   ■ A few words about Ruby
   ■ Rails' core components
   ■ RESTful architecture
2. Your first Rails application
3. Your introductory Rails exercise
What is Ruby on Rails?

Web application development framework written in Ruby

- [http://rubyonrails.org/](http://rubyonrails.org/)

Philosophy

- "Don't repeat yourself" – DRY
- Convention over Configuration – there is "the Rails way"
- RESTful architecture
- Everything in its place

- Used by Github, Groupon, Twitter (partially), openHPI
A few words about Ruby

http://www.ruby-lang.org/

- Dynamic, reflective, general-purpose, object-oriented
- Influenced by Perl, Smalltalk, Eiffel, and Lisp
- Open-source, mature software
- Matz’s Ruby Interpreter (MRI) versions:

<table>
<thead>
<tr>
<th>Version</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ruby 1.0</td>
<td>1996</td>
</tr>
<tr>
<td>Ruby 1.8.7</td>
<td>2010</td>
</tr>
<tr>
<td>Ruby 1.9.3</td>
<td>2011</td>
</tr>
<tr>
<td>Ruby 2.0.0</td>
<td>2013</td>
</tr>
<tr>
<td>Ruby 2.2.2</td>
<td>2015</td>
</tr>
</tbody>
</table>

- Additionally different VMs available (JRuby, Rubinius, IronRuby, Maglev)

Yukihiro "Matz" Matsumoto with R. Stallman

Image by Rubén Rodríguez (CC BY 3.0) - http://commons.wikimedia.org/wiki/File:Matz.jpg
Rails Core Components

**View**
- Action Pack
  - Action View (renders template)
  - Action Dispatch (parses HTTP, sessions, cookies, etc.)

**Controller**
- Action Controller (make data available, application flow)

**Model**
- Active Model (e.g. validations)
  - Active Record (ORM)
  - Database (SQL, Graph..)

**Data storage**

**Railities**
- (core code, e.g. rake)
- Active Support (utility classes, e.g. i18n)
- Action Mailer (email services)

**Gems**
- (packaged libraries)
  - [https://rubygems.org/](https://rubygems.org/)

---

Introduction to Ruby on Rails — Software Engineering II — WS 2018/19

Rails Application Layout

my_first_rails_app/
  app/
    assets/
    controller/
      application_controller.rb
    helpers/
      application_helper.rb
    models/
    views/
      application/
        index.html.erb
      layouts/
        application.html.erb
RESTful Architecture

- **Representational State Transfer (REST)** is a software architecture style for distributed systems.

- **Principles**
  - Uniform Interface
  - Stateless Interactions
  - Cacheable
  - Clients and servers
  - Layered System

- Largest RESTful implementation: World Wide Web
RESTful Architecture – HTTP verbs

- REST supports all 4 HTTP 1.1 verbs: GET, PUT, POST, DELETE
- Differentiation of collections and individual elements

<table>
<thead>
<tr>
<th>Resource</th>
<th>GET</th>
<th>PUT</th>
<th>POST</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single element</td>
<td>Retrieve</td>
<td>Update or create</td>
<td>Create</td>
<td>Delete</td>
</tr>
<tr>
<td><a href="http://localhost:3000/authors/1">http://localhost:3000/authors/1</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collection</td>
<td>List</td>
<td>Replace</td>
<td>Create</td>
<td>Delete</td>
</tr>
<tr>
<td><a href="http://localhost:3000/authors">http://localhost:3000/authors</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

25. Oktober 2018
Examples of Routes

- GET /
  # invoke “home” controller
- GET /authors
  # retrieve a list of all authors
- GET /authors/new
  # get the form to enter a new author
- POST /authors
  # create a new author
- GET /authors/1
  # show details of the first author
- GET /authors/1/edit
  # get the form to edit the first author
- PUT /authors/1
  # update the first author
- DELETE /authors/1
  # delete the first author
Introduction to Ruby on Rails

1. Ruby & Ruby on Rails
2. Your first Rails application
3. Your introductory Rails exercise
How to Start?

- **Option 1:** You use *Mac* or *Linux*
  - Install and use Ruby on Rails directly on your OS
  - Ruby version manager (e.g. RVM, rbenv) if older versions of Ruby should be kept
  - [http://guides.rubyonrails.org/getting_started.html#installing-rails](http://guides.rubyonrails.org/getting_started.html#installing-rails)
  - Or use option 2

- **Option 2:** You have *Windows* or want to use a VM *(recommended)*
  - We prepared one for you via Vagrant ([https://www.vagrantup.com/](https://www.vagrantup.com/))
  - Uses VirtualBox in the backend (free on all platforms) ([https://www.virtualbox.org/](https://www.virtualbox.org/))
  - Use your own tools & editors, run the project in a headless VM
  - See project README for setup instructions

- **Option 3:** You have *Windows* and install Ruby on Rails directly on your OS
  - Tends to consume some time, might cause problems with certain dependencies
  - [http://railsinstaller.org/en](http://railsinstaller.org/en)
Recommended to work through / read this hands-on tutorial. Seriously.
http://guides.rubyonrails.org/getting_started.html

Getting Started with Rails
This guide covers getting up and running with Ruby on Rails.
After reading this guide, you will know:
- How to install Rails, create a new Rails application, and connect your application to a database.
- The general layout of a Rails application.
- The basic principles of MVC (Model, View, Controller) and RESTful design.
- How to quickly generate the starting pieces of a Rails application.

Tip:
Before you start coding, make sure, the correct versions are installed.
$ ruby --version
$ rails --version

The following slides give a general overview
rails – Main executable

Start interactive shell to test out ideas

$ rails console

Start new rails application

$ rails new

Generate boilerplate for models, controllers & views

$ rails generate

Start the development server

$ rails server

Start a direct database shell

$ rails dbconsole

Example: generate model, controller and view without controller specs

$ rails g scaffold author last_name:string homepage:string --controller-specs false

http://guides.rubyonrails.org/command_line.html
Bundler – Ruby package manager

- Ruby libraries are packaged as "gems"
- Online repository at https://rubygems.org/
- Bundler resolves dependencies of gems
- Gemfile holds a list of required gems
  - Specify versions, e.g. `gem 'rails' >= '4.1.6'`
  - Alt. sources, e.g. `:github => "tkowark/sawyer"`
- Gemfile.lock is populated with resolved dependencies
  - Should be under version control

Manually install a gem (Ruby package)

```
$ gem install
```

Install all gems listed as dependencies in Gemfile

```
$ bundle install
```
rake – Ruby make

List all available rake commands

$ rake -T

List all configured routes

$ rake routes # Rails 4  $ rails routes # Rails 5

Setup the database and run all migrations

$ rails db:setup db:migrate

Replace database with db layout from db/schema.rb
Do not run migrations.

$ rails db:schema:load

Run Rspec (testing framework for RoR) tests

$ bundle exec rspec

or

$ rspec

Tip:
Rails 5 changed some responsibilities. Tasks might be with 'rails' instead of 'rake' now

Info:
Running schema:load is advisable when setting up a completely new project. It is not intended to work around bad migrations.
Install Git:
- `sudo apt-get install git`
- `http://git-scm.com/` (Installers for all systems)

Setting up user name and email:
- Mandatory to commit changes
- Use your github credentials!

```bash
$ git config --global user.email "vorname.nachname@student.hpi.de"
$ git config --global user.name "Max Mustermann"
```

Alternative: setting parameters only for one project:

```bash
$ cd /path/to/your/project
$ git config user.email "vorname.nachname@student.hpi.de"
$ git config user.name "Max Mustermann"
```
Git workflow – committing a change

Checkout remote repository to local copy

```bash
$ git clone https://github.com/hpi-swt2/sport-portal
```

Change main layout template app/views/layouts/application.html.erb

Stage changes (add files from working copy to repository index)

```bash
$ git add app/views/layouts
```

List changes to be committed

```bash
$ git status
```

Commit with commit messages. Reference Github issue #25

```bash
$ git commit -m "Fixed issue #25"
```

Fetch and merge changes from remote repository

```bash
$ git pull
```

Publish local commits

```bash
$ git push
```
Introduction to Ruby on Rails

1. Ruby & Ruby on Rails
2. Your first Rails application
3. Your introductory Rails exercise
4. Additional Literature
Goals

- Get familiar with Ruby (on Rails)
- Create necessary accounts for the project

Tasks

- (Create a Github account and get familiar with it)
- Visit https://classroom.github.com/a/d1BMYYkfA
- Accept the assignment
- Follow the instructions in the README file

Deadline

- Nov 9, 15:00 pm CET
- POs are not required to complete this task
Exercise – Rails Project

Accept the **test-assignment** assignment

Accepting this assignment will give your team access to the assignment repository in the [@hi-sw2-exercise](https://github.com/hi-sw2-exercise) organization on GitHub.

Please be certain that the team you are selecting is the correct team as you cannot change this later

**Join an existing team**

<table>
<thead>
<tr>
<th>Team</th>
<th>1 student</th>
<th>Join</th>
</tr>
</thead>
<tbody>
<tr>
<td>test</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**OR Create a new team**

Create a new team
Exercise – Travis CI

- Travis CI
  - Continuous Integration Server
  - Experience?

[Link to repository: hpi.swt2-exercise / rails-exercise-18-rteusner]
Exercise – Travis CI
Next Steps

- **Get to know GitHub**
  - You probably already do
  - All the cool projects are there!

- **Familiarize yourselves with Ruby on Rails and web dev**
  - Read the Ruby on Rails guide
    (https://guides.rubyonrails.org/getting_started.html)
  - Understand the core Ruby language constructs (coming from other languages)
    (https://www.ruby-lang.org/en/documentation/ruby-from-other-languages/)

- **Start the exercise once released**
Introduction to Ruby on Rails

1. Ruby & Ruby on Rails
   - What is Ruby on Rails?
   - Rails' core components
   - RESTful architecture

2. Your first Rails application
   - Folder structure
   - Databases
   - rails, git

3. Your introductory Rails exercise
   - Using Travis CI