

Table of Contents

Introduction	0
はじめに	1
Ruby on Rails入門	2
環境構築	2.1
Ruby on Rails基礎	2.2
アプリケーション開発 1	3
基本的なモデルの実装	3.1
画像アップロード(CarrierWave)	3.2
デザインの適用	3.3
アプリケーション開発 2	4
ログイン機能(Devise)	4.1
アプリケーション開発 3	5
コメント機能	5.1

Rails Training

- <https://github.com/sadah/rails-training>
- <https://sadah.github.io/rails-training/>

Install

```
npm install
```

for pdf / ebook

```
brew install caskroom/cask/calibre
```

Usage

```
npm start # http://localhost:4000
```

Build

```
npm run build
```

Tests

```
npm test
```

はじめに

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Ruby on Rails入門

- 環境構築
- Ruby on Rails基礎

Ruby on Rails 環境構築

この手順は OS X 10.8, 10.9, 10.10 を対象としています。

Xcode インストール

App Store から Xcode をインストールする。一度 Xcode を起動し、ライセンス使用許諾契約に同意する。

Command line tools インストール

Command line tools をインストールする。

```
xcode-select --install
```

Homebrew インストール

[Homebrew](#) をインストールする。

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/brew/install.sh)"
```

rbenv インストール

[rbenv](#) をインストールする。

```
brew update  
brew install rbenv rbenv-gem-rehash ruby-build
```

bashを利用している場合、以下のコマンドで環境変数を設定する。zshなどの場合、適宜書き換えて実行する。

```
echo 'eval "$(rbenv init -)"' >> ~/.bash_profile  
echo 'export PATH="$HOME/.rbenv/shims:$PATH"' >> ~/.bash_profile  
source ~/.bash_profile
```

Ruby インストール

Rubyをインストールする

```
rbenv install 2.2.3
```

デフォルトのRubyバージョンを指定する。

```
rbenv global 2.2.3
```

Ruby on Rails インストール

Ruby on Rails をインストールする。

```
gem install rails --no-ri --no-rdoc
```

確認

以下のコマンドが正しく実行できれば、Railsのインストールは正しくできています。

```
rails new sample-project
```

その他

Windows環境はサポート対象外ですが、[RailsInstaller](#) が便利だと思います。

- <https://s3.amazonaws.com/railsinstaller/Windows/railsinstaller-3.1.0.exe>

Ruby on Rails 基礎

Railsアプリケーションの作成

Railsアプリケーションを作成します。

```
rails new first-app
```

Railsアプリケーションは以下のよう構成になっています。

```
first-app
├── Gemfile # ライブラリを管理するファイル
├── Gemfile.lock
├── README.rdoc
├── Rakefile
└── app # アプリケーションのコードが配置される
    ├── assets
    │   ├── images
    │   ├── javascripts
    │   │   └── application.js
    │   └── stylesheets
    │       └── application.css
    ├── controllers
    │   ├── application_controller.rb
    │   └── concerns
    ├── helpers
    │   └── application_helper.rb
    ├── mailers
    ├── models
    │   └── concerns
    └── views
        ├── layouts
        │   └── application.html.erb
└── bin
```

```
|   └── bundle
|   └── rails
|   └── rake
|   └── setup
|   └── spring
└── config # アプリケーションの設定ファイル
    ├── application.rb
    ├── boot.rb
    ├── database.yml
    ├── environment.rb
    └── environments
        ├── development.rb
        ├── production.rb
        └── test.rb
    ├── initializers
        ├── assets.rb
        ├── backtrace_silencers.rb
        ├── cookies_serializer.rb
        ├── filter_parameter_logging.rb
        ├── inflections.rb
        ├── mime_types.rb
        ├── session_store.rb
        └── wrap_parameters.rb
    ├── locales
        └── en.yml
    ├── routes.rb
    └── secrets.yml
├── config.ru
└── db
    └── seeds.rb
└── lib
    ├── assets
    └── tasks
└── log
└── public # 静的なファイル
    ├── 404.html
    ├── 422.html
    └── 500.html
```

```
|   └── favicon.ico  
|   └── robots.txt  
└── test  
    ├── controllers  
    ├── fixtures  
    ├── helpers  
    ├── integration  
    ├── mailers  
    ├── models  
    └── test_helper.rb  
└── tmp  
    └── cache  
      └── assets  
└── vendor  
    └── assets  
      ├── javascripts  
      └── stylesheets
```

Railsアプリケーションを起動します。

```
cd first-app  
rails server
```

scaffoldによるコードの自動生成

Railsではさまざまなコードを自動生成できます。scaffoldでアプリケーションの雛形を作ります。scaffoldは一覧、詳細、追加、削除、変更のコードを自動生成します。

ここでは **bookmark** を管理する機能を作ります。

```
rails generate scaffold bookmark title:string description:text
```

DBを更新して、アプリケーションを起動します。

```
bin/rake db:migrate  
rails server
```

<http://localhost:3000/bookmarks>で動作確認ができます。

データの追加、削除、変更などを行ってみましょう。

scaffoldで生成されるファイル

scaffoldで生成されたファイルを確認していきます。scaffoldではこのような実行結果が出力されます。

```
% rails generate scaffold bookmark title:string description:text
  invoke  active_record
  create    db/migrate/20151129091144_create_bookmarks.rb
  create    app/models/bookmark.rb
  invoke  test_unit
  create    test/models/bookmark_test.rb
  create    test/fixtures/bookmarks.yml
  invoke  resource_route
    route   resources :bookmarks
  invoke  scaffold_controller
  create    app/controllers/bookmarks_controller.rb
  invoke  erb
  create    app/views/bookmarks
  create    app/views/bookmarks/index.html.erb
  create    app/views/bookmarks/edit.html.erb
  create    app/views/bookmarks/show.html.erb
  create    app/views/bookmarks/new.html.erb
  create    app/views/bookmarks/_form.html.erb
  invoke  test_unit
  create    test/controllers/bookmarks_controller_test.rb
  invoke  helper
  create    app/helpers/bookmarks_helper.rb
  invoke  test_unit
  invoke  jbuilder
  create    app/views/bookmarks/index.json.jbuilder
  create    app/views/bookmarks/show.json.jbuilder
  invoke  assets
  invoke  coffee
  create    app/assets/javascripts/bookmarks.coffee
  invoke  scss
  create    app/assets/stylesheets/bookmarks.scss
  invoke  scss
  create    app/assets/stylesheets/scaffolds.scss
```

scaffoldでは以下のようなファイルを生成しています。

- active_record
 - テーブルの作成
 - モデルの作成
- routeの設定
- controller
 - viewの作成
 - helperの作成
 - jbuilder(jsonテンプレート)の作成
- assets
 - CoffeeScriptの作成
 - Scssの作成

実際にファイルを開いて確認して見ましょう。

migration

db/migrate/20151129091144_create_bookmarks.rb

```
class CreateBookmarks < ActiveRecord::Migration
  def change
    create_table :bookmarks do |t|
      t.string :title
      t.text :description
      t.string :url

      t.timestamps null: false
    end
  end
end
```

routes

config/routes.rb

```
Rails.application.routes.draw do
```

```
resources :bookmarks

# The priority is based upon order of creation: first created
# See how all your routes lay out with "rake routes".

# You can have the root of your site routed with "root"
# root 'welcome#index'

# Example of regular route:
# get 'products/:id' => 'catalog#view'

# Example of named route that can be invoked with purchase_url
# get 'products/:id/purchase' => 'catalog#purchase', as: :pu

# Example resource route (maps HTTP verbs to controller actions)
# resources :products

# Example resource route with options:
# resources :products do
#   member do
#     get 'short'
#     post 'toggle'
#   end
#   #
#   collection do
#     get 'sold'
#   end
# end

# Example resource route with sub-resources:
# resources :products do
#   resources :comments, :sales
#   resource :seller
# end

# Example resource route with more complex sub-resources:
# resources :products do
#   resources :comments
#   resources :sales do
```

```
#       get 'recent', on: :collection
#   end
# end

# Example resource route with concerns:
# concern :toggleable do
#   post 'toggle'
# end
# resources :posts, concerns: :toggleable
# resources :photos, concerns: :toggleable

# Example resource route within a namespace:
# namespace :admin do
#   # Directs /admin/products/* to Admin::ProductsController
#   # (app/controllers/admin/products_controller.rb)
#   resources :products
# end
end
```

models

app/models/bookmark.rb

```
class Bookmark < ActiveRecord::Base
end
```

controllers

app/controllers/bookmarks_controller.rb

```
class BookmarksController < ApplicationController
  before_action :set_bookmark, only: [:show, :edit, :update, :de
  # GET /bookmarks
```

```
# GET /bookmarks.json
def index
  @bookmarks = Bookmark.all
end

# GET /bookmarks/1
# GET /bookmarks/1.json
def show
end

# GET /bookmarks/new
def new
  @bookmark = Bookmark.new
end

# GET /bookmarks/1/edit
def edit
end

# POST /bookmarks
# POST /bookmarks.json
def create
  @bookmark = Bookmark.new(bookmark_params)

  respond_to do |format|
    if @bookmark.save
      format.html { redirect_to @bookmark, notice: 'Bookmark was successfully created.' }
      format.json { render :show, status: :created, location: @bookmark }
    else
      format.html { render :new }
      format.json { render json: @bookmark.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /bookmarks/1
# PATCH/PUT /bookmarks/1.json
def update
```

```
respond_to do |format|
  if @bookmark.update(bookmark_params)
    format.html { redirect_to @bookmark, notice: 'Bookmark was successfully updated.' }
    format.json { render :show, status: :ok, location: @bookmark }
  else
    format.html { render :edit }
    format.json { render json: @bookmark.errors, status: :unprocessable_entity }
  end
end

# DELETE /bookmarks/1
# DELETE /bookmarks/1.json
def destroy
  @bookmark.destroy
  respond_to do |format|
    format.html { redirect_to bookmarks_url, notice: 'Bookmark was successfully destroyed.' }
    format.json { head :no_content }
  end
end

private
# Use callbacks to share common setup or constraints between actions
def set_bookmark
  @bookmark = Bookmark.find(params[:id])
end

# Never trust parameters from the scary internet, only allow the data we
# know is safe.
def bookmark_params
  params.require(:bookmark).permit(:title, :description, :url)
end
```

helpers

app/helpers/bookmarks_helper.rb

```
module BookmarksHelper  
end
```

views

app/views/bookmarks/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Listing Bookmarks</h1>

<table>
  <thead>
    <tr>
      <th>Title</th>
      <th>Description</th>
      <th>Url</th>
      <th colspan="3"></th>
    </tr>
  </thead>

  <tbody>
    <% @bookmarks.each do |bookmark| %>
      <tr>
        <td><%= bookmark.title %></td>
        <td><%= bookmark.description %></td>
        <td><%= bookmark.url %></td>
        <td><%= link_to 'Show', bookmark %></td>
        <td><%= link_to 'Edit', edit_bookmark_path(bookmark) %></td>
        <td><%= link_to 'Destroy', bookmark, method: :delete, data: { confirm: 'Are you sure?' } %></td>
      </tr>
    <% end %>
  </tbody>
</table>

<br>

<%= link_to 'New Bookmark', new_bookmark_path %>
```

app/views/bookmarks/_form.html.erb

```
<%= form_for(@bookmark) do |f| %>
<% if @bookmark.errors.any? %>
<div id="error_explanation">
<h2><%= pluralize(@bookmark.errors.count, "error") %> proh
<ul>
<% @bookmark.errors.full_messages.each do |message| %>
<li><%= message %></li>
<% end %>
</ul>
</div>
<% end %>

<div class="field">
<%= f.label :title %><br>
<%= f.text_field :title %>
</div>
<div class="field">
<%= f.label :description %><br>
<%= f.text_area :description %>
</div>
<div class="field">
<%= f.label :url %><br>
<%= f.text_field :url %>
</div>
<div class="actions">
<%= f.submit %>
</div>
<% end %>
```

app/views/bookmarks/index.json.jbuilder

```
json.array!(@bookmarks) do |bookmark|
  json.extract! bookmark, :id, :title, :description, :url
  json.url bookmark_url(bookmark, format: :json)
end
```

課題

1. `link_to`を使ってURLをリンクにしてみましょう。
2. タイトルとURLを必須項目にしましょう。