
Update aplikacji z Rails 3 do Rails 4 - Tutorial

Tomasz Sott

Spis treści

| | |
|---|----|
| Wstęp | 3 |
| Zmiany w Gemfile | 4 |
| Zmiany w config/application.rb | 5 |
| Zmiany w config/routes.rb | 6 |
| Zmiany w config/environments development.rb, production.rb, test.rb | 7 |
| Zmiany w config/initializers/secret_token.rb | 8 |
| Zmiany w app/post.rb oraz app/controllers/posts_controller.rb | 9 |
| Bibliografia | 11 |

Wstęp

Tutorial ten ma na celu pokazanie zmian jakie należy dokonać przy zmianie aplikacji opartej o Rails 3 na Rails 4. Do przejścia na Rails 4 wymagane jest aby aplikacja była wykonana w Rails wersji co najmniej 3.2.

Tutorial oparty jest na aplikacji Księga pamiątkowa.

Aplikacja dostępna jest pod adresem:

<http://herokuapp.com/ksiega-pamiatkowa>.

Kod źródłowy do wersji 3.2 oraz 4.0 znajduje się na:

http://github.com/tsott/arch_serw_int

Zmiany w Gemfile

Przejście z Rails 3 do Rails 4 należy zacząć od zmian w pliku Gemfile.

Oczywistą zmianą jest zmiana gem'u rails 3.2.8 na rails 4.0.0.rc1. Trzeba także zmienić gem'y **sass-rails** (z 3.2.3 na 4.0.0.rc1), **coffee-rails** (z 3.2.1 na 4.0.0)

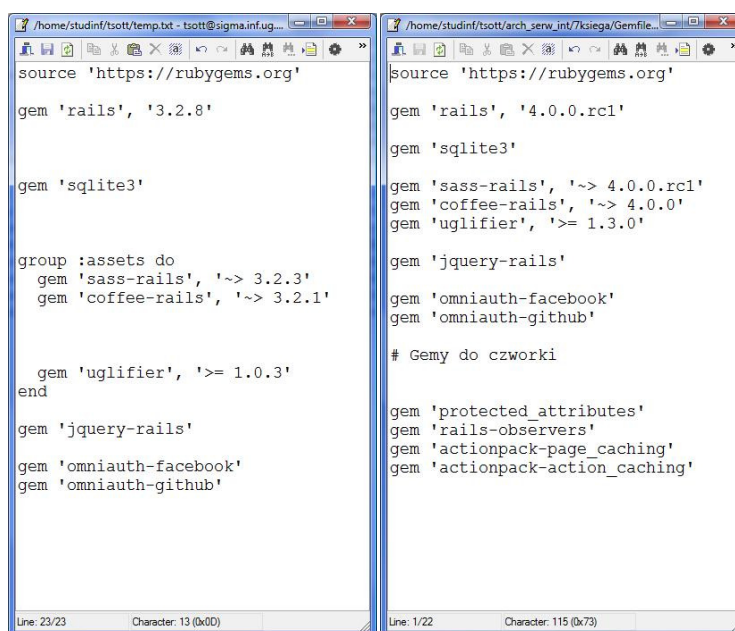
oraz **uglifier** (z 1.0.3 na 1.3.0) Po dokonaniu tych zmian wykonujemy

polecenie **bundle update**. Następnie aby pozostałe użyte gem'y pasowały do nowej wersji aplikacji wykonujemy polecenie **bundle outdated**.

Polecenie to pokazuje aktualne wersje gem'ow i wersje które powinny zostać użyte by zachować zgodność z Rails 4. W nowej wersji Rails nie ma w Gemfile grupy assets, dlatego też należy usunąć z pliku linijki za to odpowiadające W celu dodatkowego uproszczenia przejścia na rails 4 należy do pliku Gemfile dodać następujące gem'y:

```
gem 'protected_attributes'
gem 'rails-observers'
gem 'actionpack-page_caching'
gem 'actionpack-action_caching'
```

Poniżej przedstawiony został plik Gemfile przed zmianami oraz po ich wprowadzeniu.



Po dokonaniu zmian wykonujemy polecenie **bundle update**.

Zmiany w config/application.rb

Rails 3 wymagało załadowania gemów pasujących do grup assets w środowiskach development i tests. Fragment kodu za to odpowiedzialny znajduje się w pliku config/application.rb i wygląda następująco:

```
if defined?(Bundler)
  # If you precompile assets before deploying to production, use this line
  Bundler.require(*Rails.groups(:assets => %w(development test)))
  # If you want your assets lazily compiled in production, use this line
  # Bundler.require(:default, :assets, Rails.env)
end
```

W rails 4 zostało to uproszczone. Powyższy fragment kodu należy zastąpić linią:

```
Bundler.require(:default, Rails.env)
```

W opisywanym pliku należy także usunąć następujące linie:

```
config.active_record.whitelist_attributes = false
config.assets.enabled = true
```

Zmiany w config/routes.rb

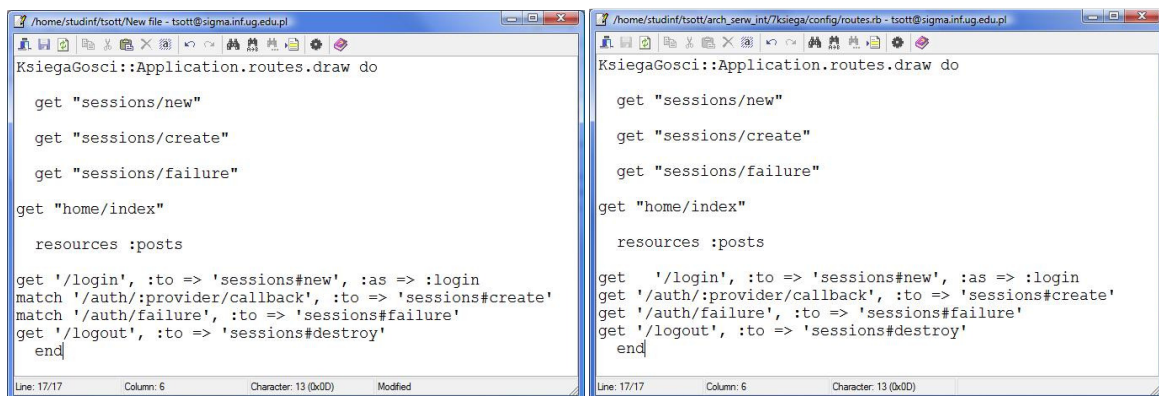
W pliku routes.rb należy zmienić wszystkie wystąpienia metody `match`.

Można to zrobić na dwa sposoby. Pierwszy z nich to zamiana metody `match` na metodę `get` (Sposób zaprezentowany na poniższych zrzutach ekranu).

Drugi sposób to dopisanie możliwych metod do każdego wystąpienia metody `match`. Możliwe do dopisania są tablica metod lub wszystkie możliwe metody. Przykładowo:

```
match '/auth/:provider/callback', :to => 'sessions#create', via: :all
```

Plik routes.rb przed oraz po dokonanych zmianach:



Zmiany w config/environments development.rb, production.rb, test.rb

Po wykonaniu powyższych kroków aplikacja uruchamia się. W konsoli wyświetlają się jednak komunikaty o przestarzałym zastosowaniu [whiny_nils](#). W celu usunięcia tych komunikatów należy w plikach development.rb, production.rb oraz test.rb usunąć linię kodu:

```
config.whiny_nils = true
```

Należy zastąpić ją w plikach development.rb i test.rb następującą linią:

```
config.eager_load = false
```

W pliku production.rb wartość [config.eager_load](#) powinna być ustawiona na [true](#). Należy zamienić linię dotyczącą kompresji:

```
config.assets.compress = true
```

Na:

```
config.assets.js_compressor = :uglifier
```

Dodatkowo w pliku development.rb należy usunąć:

```
config.action_dispatch.best_standards_support = :builtin  
config.active_record.mass_assignment_sanitizer = :strict  
config.active_record.auto_explain_threshold_in_seconds = 0.5  
config.assets.compress = false
```

W pliku test.rb usunięta powinna zostać linia:

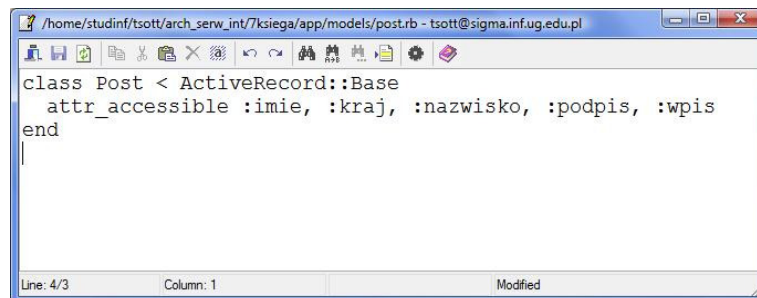
```
config.active_record.mass_assignment_sanitizer = :strict
```

Zmiany w config/initializers/secret_token.rb

W Rails 4 zmianie ulega opcja konfiguracji `Application.config.secret_token`. Zastępuje ją `Application.config.secret_key_base`. Zaleca się pozostawienie obydwu linii do czasu całkowitego przejścia na Rails 4. Zaleca się także zmianę wartości `Application.config.secret_key_base` tak by różniła się od wartości `Application.config.secret_token`. Zmiana ta jest konieczna, ponieważ ciasteczko sesji zostaje zaszyfrowane tak by użytkownik nie mógł w prosty sposób zobaczyć jego zawartości.

Zmiany w app/post.rb oraz app/controllers/posts_controller.rb

W celu zmiany tzw strong parameters należy atrybuty przenieść z modelu do kontrolera. Poniższy zrzut ekranu pokazuje plik post.rb. Usuwamy z niego linię kodu zawierającą atrybuty.

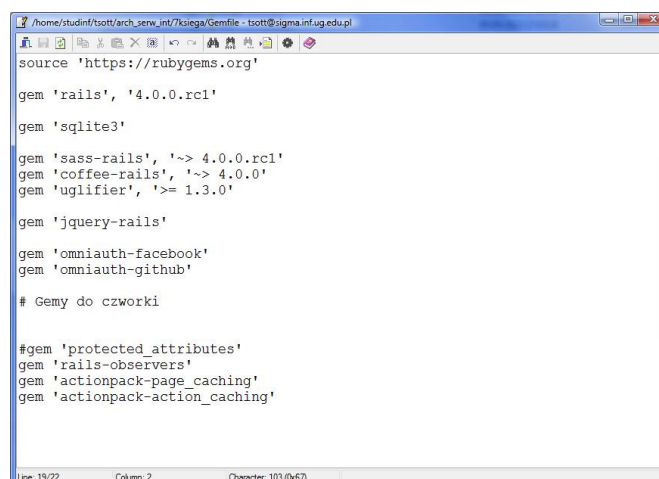


```
class Post < ActiveRecord::Base
  attr_accessible :imie, :kraj, :nazwisko, :podpis, :wpis
end
```

Atrybuty przenosimy do pliku posts_controller.rb gdzie tworzymy następujący fragment kodu:

```
def post_params
  params.require(:post).permit(:imie, :kraj, :nazwisko, :podpis, :wpis)
end
```

Następnie w pliku posts_controller.rb zastępujemy wszystkie wystąpienia `params[:post]` przez `post_params`. Po wykonaniu tego można z pliku Gemfile usunąć gem `protected_attributes`, ponieważ nie jest już potrzebny. Plik Gemfile powinien wyglądać następująco:



```
source 'https://rubygems.org'

gem 'rails', '4.0.0.rc1'
gem 'sqlite3'

gem 'sass-rails', '~> 4.0.0.rc1'
gem 'coffee-rails', '~> 4.0.0'
gem 'uglifier', '>= 1.3.0'

gem 'jquery-rails'

gem 'omniauth-facebook'
gem 'omniauth-github'

# Gemy do czworki

#gem 'protected_attributes'
gem 'rails-observers'
gem 'actionpack-page_caching'
gem 'actionpack-action_caching'
```

Ostatnim etapem jest usunięcie folderu plugins z folderu vendors. Folder vendors w Rails 4 nie jest używany. Wtyczki należy przenieść do folderu lib lub umieścić jako gem'y. Po dokonaniu zmian opisanych w tym tutorialu aplikacja początkowo oparta na Rails 3 powinna działać w Rails 4.

Bibliografia

- #415 Upgrading to rails 4 - RailsCasts
<http://railscasts.com/episodes/415-upgrading-to-rails-4?autoplay=true>
- Ruby on Rails guides
http://edgeguides.rubyonrails.org/upgrading_ruby_on_rails.html
- Upgrading to Rails 4 - Parameters Security Tour
<http://iconoclastlabs.com/cms/blog/posts/upgrading-to-rails-4-parameters-security-tour>