MIT CD

Messaging in Ruby mit RabbitMQ ▶ 49

S&S

Ruby

Ausgabe Januar/Februar 1.10

RailsWay

Das Magazin für Ruby on Rails

www.railsway-magazin.de

CD-Inhalt

RailsWayTV

Neal Ford auf der RailsWayCon 2009: „Metaprogramming Ruby for Fun & Profit“

Ruby

Ruby Enterprise Edition, JRuby 1.4, Rails 2.3.4, Ruby Sources

Open-Source-Tools

Apache ActiveMQ 5.3, Tsung 1.3.1, FXRuby 1.6.20, Ruby-GPGME 1.0.8, Rake 0.8.7, Mongrel 1.1.5

Mehr Infos auf Seite 3

Enterprise Rails

Rails Apps im JBoss-Server ▶ 78

Paperclip

Einführung in das beliebte

innoQ

Sonderdruck der Firma innoQ

ActiveMerchant

PayPal in Rails Apps integrieren ▶ 62

DataMapper

Die alternative ORM-Lösung ▶ 38

Gemcutter vs. RubyForge

Die Schmiede kühlt ab ▶ 74

Factory Girl

Testdaten aus der Fabrik ▶ 28

Shoulda

Einstieg ins Behaviour-driven Development ▶ 57

Datenträger enthält Info- und Lehrprogramme gemäß §14 JuSchG

Formulare endlich einfach

Fantastische Formulare

Robert Glaser

Formulare sind sicherlich das Stiefkind im Alltag eines Rails-Entwicklers: Felder für die verschiedenen Attribute anlegen, Standardwerte und Vorauswahlen einbauen, Fehlermeldungen und Hinweisfelder hinzufügen und letztendlich das Ganze noch in brauchbares semantisches HTML verpacken - Sie kennen das sicher zur Genüge. Das i-Tüpfelchen: Die meisten Webanwendungen sind voll von Formularen!

Das dachte sich sicherlich auch Justin French und entwickelte schlussendlich das Gem „Formtastic" samt einer speziellen DSL (Domain Specific Language), das sich anschickt, das tägliche Bauen von Webformularen in einer Rails-Anwendung gründlich zu vereinfachen. Die Lösung, die Formtastic bietet, ist sicherlich keine Eintagsfliege: Das bekannte Rails-Consulting-Unternehmen Thoughtbot baut mittlerweile alle Formulare mit Formtastic.

Installation und Vorbereitungen

Formtastic wird als Gem bereitgestellt und lässt sich daher wie gewohnt installieren. Falls Sie Ihrer Rubygems-Installation noch nicht das neue Standard Gem Repository „Gemcutter" hinzugefügt haben, sollten Sie das vor der Installation von Formtastic noch kurz nachholen mittels *$ sudo gem sources -a http://gemcutter.org*. Nun können Sie das Gem wie gewohnt installieren: *$ sudo gem install formtastic*. Im Anschluss teilen Sie der Rails-Konfiguration Ihrer Anwendung noch mit, dass das Formtastic Gem benötigt wird. Dazu fügen Sie in den *config*-Block der *environment.rb* folgende Zeile hinzu: *config.gem 'formtastic'*. Schlussendlich sollten Sie in jedem Fall noch im Verzeichnis Ihrer Anwendung den Befehl *$ script/generate formtastic* ausführen. Dadurch wird in *config/initializers* eine Konfigurationsdatei erstellt, in der Sie später evtl. nötige Einstellungen für die Bibliothek vornehmen können – außerdem werden zwei CSS-Dateien in *public/stylesheets* generiert: *formtastic.css* und *formtastic_changes.css*. In der ersten sollten Sie nichts ändern, die zweite ist speziell für gewünschte CSS-Modifikationen da. Nun stellen Sie abschließend noch sicher, dass die neuen CSS-Dateien auch in Ihr Layout-File eingebunden werden: *<%= stylesheet_link_tag 'formtastic', 'formtastic-changes' %>*. Wichtig: Achten Sie darauf, dass Ihr HTML Header valide ist und das *<html>*-Element ein *xmlns*-Attribut mitbringt – ansonsten werden die Formtastic-CSS-Definitionen nicht angezogen werden.

kurz und bündig

Inhalt

Unterstützung für Formulare und assoziierte Modelle für Rails

Quellcode mit angeliefert

Nein

Ausgangssituation

Gehen wir davon aus, dass wir für unser Beispiel zwei einfache Modelle *Book* und *Author* mit einer Relation in unserer Anwendung haben:

```
class Book < ActiveRecord::Base
  belongs_to :author
end

class Author < ActiveRecord::Base
  has_many :books
end
```

Der Fall „einfach und schnell"

Wenn man nun einfach mal die Features von Formtastic ausprobieren will oder beispielsweise nur ein Pflegeformular für einen geschützten Bereich entwickeln möchte, das keine speziellen Felder ausblenden muss, kann man einfach für jedes Attribut eines

Modells die Formularfelder erkennen und generieren lassen – das umfasst auch die obligatorischen Buttons:

```
<% semantic_form_for @book do |form| %>
  <%= form.inputs %>
  <%= form.buttons %>
<% end %>
```

Wie man sieht, ist die Syntax den bekannten Rails-Formular-Helper-Methoden sehr ähnlich: Statt eines *form_for*-Aufrufs nutzt man einfach die von Formtastic bereitgestellte Methode *semantic_form_for*. Wenn Sie nun Ihre Formularseite aufrufen, werden Sie sehen, dass Formtastic automatisch ein Formular für alle Attribute des übergebenen Modells generiert und ebenfalls in ein semantisch korrektes HTML-Konstrukt eingebettet hat. Dadurch, dass Sie zu Anfang die CSS-Dateien eingebunden haben, sollte nun ebenfalls die Ausrichtung und das Formularlayout sauber dargestellt werden. Abhängig von Ihrem Datenmodell werden Sie feststellen, dass Formtastic außerdem etwaige Relationen aufgelöst hat und auch für die referenzierten Modelle Felder bereitstellt!

Der Fall „etwas mehr Anpassung, bitte!“

Die meisten Fälle werden realistischerweise etwas mehr Anpassung beim Erstellen eines Formulars erfordern – die Formtastic-DSL bietet hierzu mehr als genug Möglichkeiten. Sollen beispielsweise nur bestimmte Felder eingeblendet oder auch ein anderes Input-Element genutzt werden, ist auch das möglich:

```
<% semantic_form_for @book do |form| %>
  <% form.inputs do %>
    <%= form.input :title, :required => true %>
    <%= form.input :description, :input_html => { :size => 60 } %>
    <%= form.inputs :first_name, :last_name, :for => :author, :name => "Author" %>
    <%= form.input :published_at, :start_year => 1900 %>
  <% end %>
<% end %>
```

Hier sehen wir, wie nur ausgewählte Attribute im Formular Felder erhalten und wie man Formtastic Feldanpassungen übergeben kann. Eine vollständige Auflistung aller Möglichkeiten für jedes Formularelement findet sich in der Formtastic-Dokumentation, wir wollen hier nur beispielhaft auf einige Möglichkeiten eingehen. Standardmäßig erwartet Formtastic für Pflichtfelder den Parameter *:required* – diese Notwendigkeit entfällt, wenn Sie zusätzlich das Plug-in *validation_reflection* (*http://github.com/redinger/validation_reflection*) installieren, damit Formtastic Zugriff auf die in Ihrem Model definierten Rails-Validierungen erhält. Das Vorhandensein des Plug-ins erkennt Formtastic im Übrigen automatisch. Mittels des Parameters *:input_html* teilen Sie für das jeweilige Input-Element die entsprechenden HTML-Eigenschaften mit. Ebenfalls sehen Sie, dass wir sehr einfach referenzierte Modelle erzeugen können – der Autor für das Buch, das wir mit dem Beispielformular anlegen wollen, kann also automatisch mit erzeugt werden. Diese Funktion setzt natürlich Rails 2.3 voraus, da hierin erstmals Nested Forms unterstützt werden.

Fehlermeldungen und Konfiguration

Standardmäßig stellt Formtastic Fehlermeldungen direkt neben dem fehlerhaften Feld „inline“ dar. Wer lieber den klassischen Rails-Weg der gesammelten Fehlermeldungen oder etwas ganz anderes bevorzugt, kann unter *config/initializers/formtastic.rb* folgende Zeile hinzufügen: *Formtastic::SemanticFormBuilder.inline_errors = :none*. Generell bietet Formtastic sehr viele Konfigurationsmöglichkeiten, die auch alle in der Dokumentation gut erklärt sind. Zusätzlich gibt es in den Formtastic Templates eine Beispielkonfigurationsdatei (http://github.com/justinfrench/formtastic/blob/master/generators/formtastic/templates/formtastic.rb).

Internationalisierung

„Last but not least“ mangelt es der Bibliothek auch nicht an Möglichkeiten zur Internationalisierung. Dies bietet sich nur an, da Formtastic gerne etliche Strings in den Templates generiert, um dem Entwickler Arbeit abzunehmen – als Beispiel seien nur die automatisch generierten Labels genannt. Standardmäßig bietet sich natürlich die Möglichkeit an, für jedes Formularelement einfach die Übersetzung des Labels manuell zu übergeben:

```
<% semantic_form_for @book do |form| %>
  <%= form.input :title, :label => I18n.t('activerecord.attributes.user.title') 
<% end %>
```

Wie Sie sehen, erzeugt das bei vielen Feldern und mehreren Formularen schon nach kurzer Zeit erhebliche Codemengen, die sicherlich unnötig sind. Formtastic stellt zu diesem Zweck ein Internationalisierungs-API bereit. Um dieses zu nutzen, müssen Sie lediglich in der Konfigurationsdatei den automatischen Lookup für Übersetzungen aktivieren: *Formtastic::SemanticFormBuilder.i18n_lookups_by_default = true*. Nun können Sie auf dem gewohnten Weg in Ihren Locale-Dateien (*config/locales*) Übersetzungen definieren:

```
de:
  formtastic:
    labels:
      book:
        title: "Titel des Buchs"
        description: "Kurzbeschreibung"
```

Danach müssen Sie lediglich den gewählten String als Label-Parameter übergeben, damit die Übersetzung gesucht werden kann. Ein Extra-Aufruf von *I18n.t* wird nicht mehr benötigt. In der Dokumentation finden Sie außerdem noch etliche sehr viel weiter gehende Möglichkeiten für die Internationalisierung Ihrer Formulare.

Links & Literatur

[1] Formtastic bei github: http://github.com/justinfrench/formtastic
[2] Google Group: http://groups.google.com.au/group/formtastic
[3] Dokumentation: http://rdoc.info/projects/justinfrench/formtastic
[4] TextMate Bundle: http://github.com/grimen/formtastic_tmbundle

Robert Glaser arbeitet seit über 3 Jahren mit Rails und ist Consultant bei der innoQ Deutschland GmbH. In seiner Freizeit widmet er sich der Produktion und dem Auflegen von elektronischer Musik.