

# Rails 2

*New features for your applications*

by Ryan Daigle Traduit par Frailers.net

## CONTENTS

<b>4</b>	<b>ActiveRecord</b>		
4	Validations		
6	Cache de requêtes		
9	Migrations Sexy		
11	Foxy Fixtures		
<b>18</b>	<b>ActionController &amp; ActionView</b>		
18	Serveurs de contenus statiques		
21	Cache des contenus statiques		
24	Sessions basées sur cookie par défaut		
26	Mises en page de partiels		
30	Mise à jour du routage RESTful		
33	Traitements des exceptions avec rescue_from		
36	Authentification HTTP		
38	Accès aux assistants facilité		
<b>40</b>	<b>Divers</b>		
40	Nettoyez votre fichier d'environnement		
41	Collections de fixtures		
42	Tâches Rake		
47	Mise à jour de script/generate		
47	Déboguage		
<b>53</b>	<b>Fonctionnalités dépréciées</b>		
53	Dépréciations provenant de versions précédentes		
57	Suppression de la pagination		
58	Plugins acts_as		
59	Suppression des adaptateurs de bases de données		
59	smtp_settings		
60	Routes renommées		
<b>62</b>	<b>Changelogs</b>		
62	ActionMailer		
63	ActionPack		
112	ActiveRecord		
142	ActiveSupport		
<b>151</b>	<b>Revisions</b>		

## INTRODUCTION

Les personnes ayant vu la présentation (<http://www.bestechvideos.com/2007/10/02/david-heinemeier-hansson-keynote-at-railsconf-2007>) de David Heinemeier Hansson (<http://www.loudthinking.com>) à RailsConf 2007 savent que Rails 2.0 ne contient pas des vagues de nouvelles fonctionnalités extraordinaires et de changements de paradigme. Au contraire, il se concentre sur un meilleur support des architectures web RESTful, sur l'aboutissement de quelques petites fonctionnalités et sur la suppression de bibliothèques et fonctionnalités qui ont été jugées non essentielles pour Rails.

Cela fait plusieurs versions que j'ai couvert les nouvelles fonctionnalités de Rails, au fur et à mesure qu'elles ont été introduites, dans la série d'articles What's New in Edge Rails (<http://ryandaigle.com>) (*Quoi de neuf dans Rails Edge*). La plupart des fonctionnalités majeures y ont été discutées, mais certaines non et certaines ont même légèrement changé depuis leur première soumission. J'espère que ce mini-livre fournira aux développeurs un support opportun résumant de manière claire les nouvelles fonctionnalités et comment les utiliser.

Ryan Daigle, *fondateur*, yFactorial, LLC (<http://yfactorial.com>)

# ActiveRecord

## CHAPTER 1

ActiveRecord est la partie la plus importante de Rails, il n'est donc pas surprenant qu'elle ait bénéficié d'améliorations dans Rails 2.0.

## Validations

### Les validations numériques deviennent utiles

La méthode `validates_numericality_of` semble être une validation vraiment utile, et pourtant elle m'a déçu plus d'une fois. Je pense que ce ne sera plus le cas avec les nouvelles options proposées dans Rails 2.

Voici un rapide aperçu de ces options:

```

validations/numericality.rb
validates_numericality_of :salary,
                          :greater_than => 39999

validates_numericality_of :salary,
                          :greater_than_or_equal_to => 40000

validates_numericality_of :ten,
                          :equal_to => 10

validates_numericality_of :bonus,
                          :less_than => 5000

validates_numericality_of :bonus,
                          :less_than_or_equal_to => 4999

validates_numericality_of :prime,
                          :odd => true

```

#### PORTÉE DU CACHE

Le cache de requêtes est éphémère. Comme Rails est "single-threaded", le cache vit tant que l'instance du modèle hôte vit. Dans la plupart des cas, c'est juste le temps de traiter la requête HTTP entrante. Le résultat est que des requêtes select identiques lors de différentes requêtes HTTP ne seront pas en mesure de tirer avantage du cache de requêtes.

```
validates_numericality_of :squared,
                          :even => true
```

Vous pouvez même combiner les conditions (assurez-vous juste qu'elles n'entrent pas en conflit):

```
validates_numericality_of :salary,
                          :greater_than_or_equal_to => 8000,
                          :even => true
```

## :allow\_blank

Avez-vous déjà utilisé l'option :allow\_nil pour sauter une validation si l'attribut en question est nil?

```
ar/post.rb
class Post < ActiveRecord::Base
  validates_numericality_of :rating, :allow_nil => true
end

Post.new(:rating => nil).valid?
#=> true
```

Ceci est utile pour les attributs optionnels ou pour les attributs dont l'existence est déjà validée par `validates_presence_of`. Mais pour les attributs optionnels qui proviennent de vos requêtes standards, la valeur est souvent une chaîne vide (étant donné que le champ a été envoyé avec une valeur vide). Auparavant, la seule manière de gérer ce cas était de créer votre propre proc `:if` :

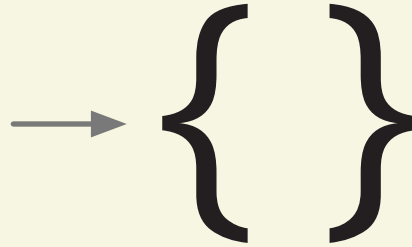
```
validations/if_proc.rb
class Post < ActiveRecord::Base

  validates_numericality_of :rating,
    :if => Proc.new { |post| not post.rating.blank? }
```

# LE CACHE DE REQUÊTES

① Une requête SQL est créée... ...le cache est vide...

```
SELECT *  
FROM users  
WHERE id = 7
```



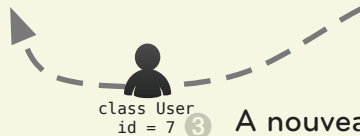
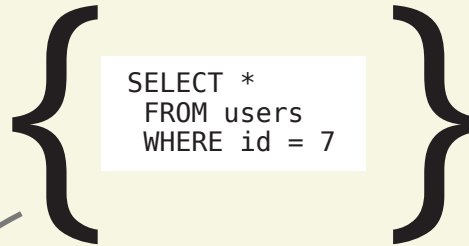
...donc la base de données est cherchée et l'enregistrement mis en cache.



② Plus tard dans la requête HTTP, la même requête est créée...

...elle est trouvée dans le cache, donc l'objet User est renvoyé immédiatement.

```
SELECT *  
FROM users  
WHERE id = 7
```



③ A nouveau durant la même requête HTTP, un enregistrement est mis à jour...

...l'entiereté du cache est vidé.

```
UPDATE orders  
SET (amount = 100)  
WHERE id = 38273
```



PeepCode: Rails 2 PDF

©2008 Ryan Daigle

Every effort was made to provide accurate information in this document. However, neither Ryan Daigle nor Topfunky Corporation shall have any liability for any errors in the code or descriptions presented in this book.

“Rails” and “Ruby on Rails” are trademarks of David Heinemeier Hansson.

This document is available for single users for US\$9 at PeepCode.com (<http://peepcode.com>). Group discounts and site licenses can also be purchased by sending email to [peepcode@topfunky.com](mailto:peepcode@topfunky.com).

## OTHER PEEPCODE PRODUCTS

- RSpec (<http://peepcode.com/products/rspec-basics>) – A three part series on the popular behavior-driven development framework.
- Rails from Scratch (<http://peepcode.com>) – Learn Rails!
- RESTful Rails (<http://peepcode.com/products/restful-rails>) – Teaches the concepts of application design with REST.
- Subscription pack of 10 (<http://peepcode.com/products/subscription-pack-of-10>) – Save money! Buy 10 PeepCode credits.
- Javascript with Prototype (<http://peepcode.com/products/javascript-with-prototypejs>) – Code confidently with Javascript!
- Rails Code Review PDF (<http://peepcode.com/products/draft-rails-code-review-pdf>) – Common mistakes in Rails applications, and how to fix them.