Ruby, Ruby on Rails

Cyril Mougel

25 novembre 2008

L'historique

- Créé en 1993 par Yukihiro Matsumoto dit « Matz »
- Langage de scripting de haut niveau
- Plus qu'orienté objet : tout est objet
- Applique le principe de moindre surprise (POLS, *principle of least surprise*)
- Fonctionne sur les plateformes les plus populaires du marché (Linux, Windows, Mac)

Le framework Ruby on Rails

- Créé par David Heinemeier Hansson dit « DHH »
- Extrait de l'application BaseCamp de 37signals
- Première *public release* en 2004

Pourquoi Rails en Ruby ?

- Multi-plateformes
- Fortes facilités d'introspection et de réflexion
  - User.find_by_firstname_and_lastname 'David', 'Hanson'
  - has_many :galleries

Le concept de Ruby on Rails

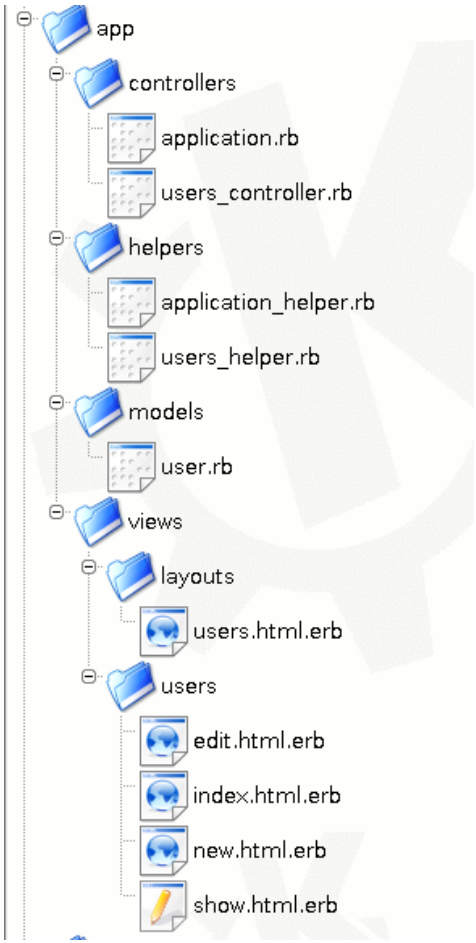
- Ruby on Rails est conçu par des développeurs, pour des développeurs
- Un cadre de travail minimal et complet pour le développement Web
- *Convention over Configuration*
- *Don't Repeat Yourself* (DRY)
- *Say what you do, Do what you say*
- Un seul et unique langage pour tout faire

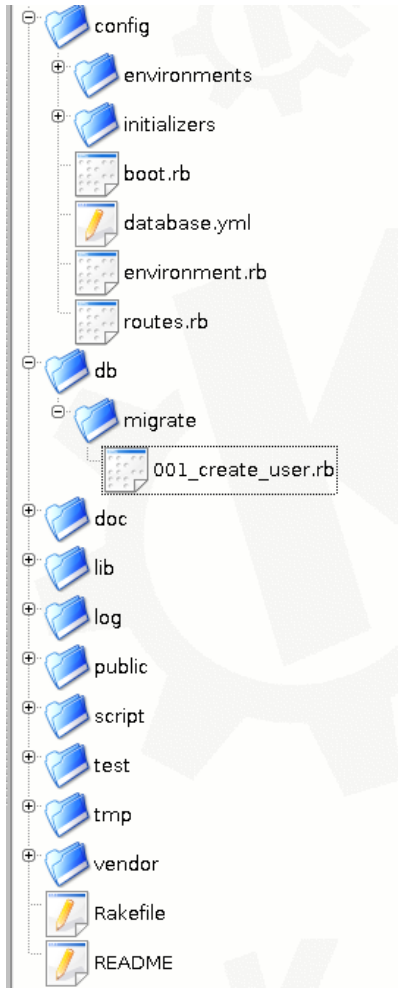
Rails est MVC

- Model : ActiveRecord
- View : ActionView
- Controller : ActionController

Chaque chose à sa place

Chaque dossier correspond à quelque chose et a son utilité propre





Gestion de la Base de données

- Système de gestion de migration(ActiveRecord : :Migration)
- Utilisation du pattern ActiveRecord
- Génération de nombreuses méthodes utilitaires à la volée

Migration de Base de donnée

- Gestion incrémentale des fichiers de migrations
- Retour avant/arrière au sein des migrations
- Utilisation de méthodes Ruby au lieu des requêtes SQL pur

Exemple de Migration

Un exemple de fichier de migration :

```
1  class FixProfiles < ActiveRecord::Migration
2    def self.up
3      create_table :products do |p|
4        p.string :titre, :limit => 100
5      end
6
7      remove_column :users, :profile_id
8      add_column :users, :profile_id, :integer
9      admin_id = Profile.find_by_label('admin').id
10     User.update_all("profile_id = #{admin_id}")
11   end
12
13   def self.down
14     drop_table :products
15
16     remove_column :users, :profile_id
17     add_column :users, :profile_id, :integer, :default => 1
18   end
19 end
```

Utilisation d'ActiveRecord

Exemple de fichier qui mappe la table Projects qui possèdent 6 champs :

Code de la classe de mapping

```
1  class Project < ActiveRecord::Base
2  end
```

Exemple d'utilisation de la classe de mapping

```
1  project1 = Project.new
2  project1.title = 'EDF Entreprise'
3  project1.save
4
5  project2 = Project.create(:title => 'GARI')
6  project2.destroy
7
8  all_Projects = Project.find :all
```

les méthodes de classe (correspondant au static de Java) accessibles pour la classe Project

Et bien-sûr toutes les méthodes accessibles en directement par la Classe :

- Project.find :all
- Project.find_by_title 'EDF Entreprise'
- Project.find_by_description 'un site'
- Project.find_by_title_and_description 'EDF', 'site'

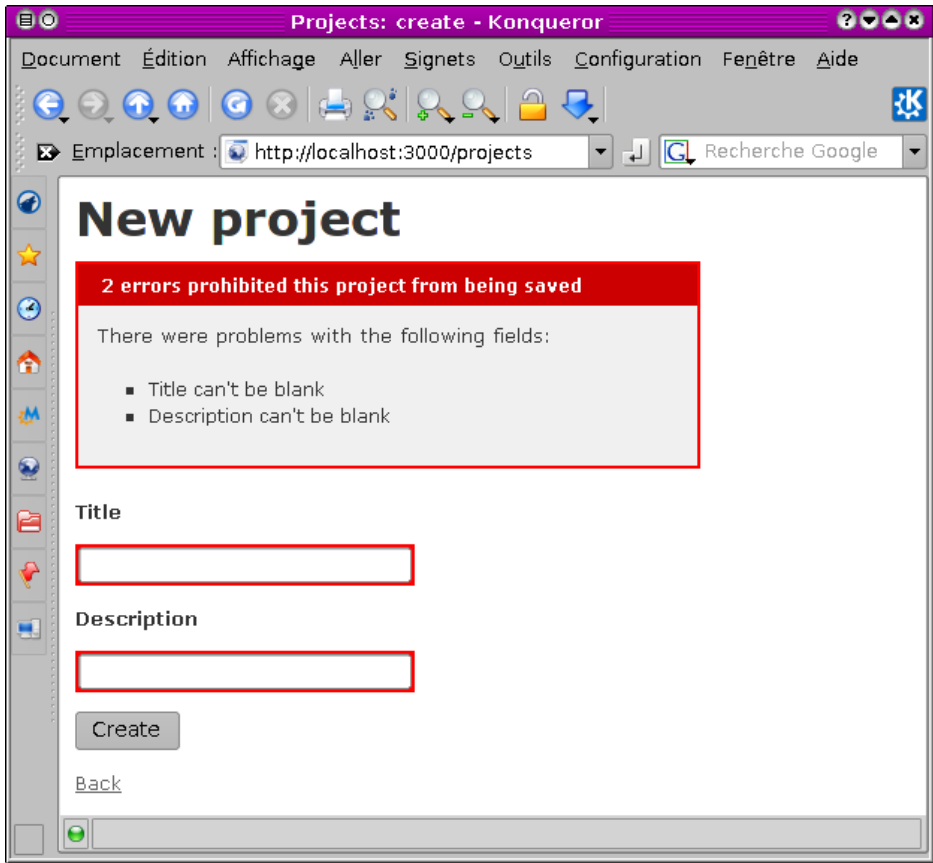
etc. . .

- Project.count :all
- Project.count_by_title 'EDF Entreprise'
- Project.count_by_description 'un site'
- Project.count_by_title_and_description 'EDF', 'site'

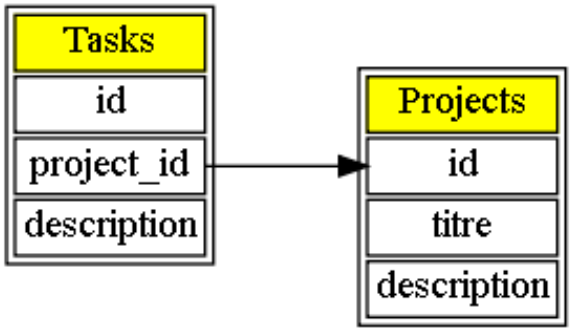
le système de validation du modèle

Multiples systèmes de gestion directement dans le cycle de Vie de l'objet modéle pour empêcher l'enregistrement en base de données d'informations erronées

```
1  class Project < ActiveRecord::Base
2    validates_presence_of :title
3    validates_presence_of :description
4    validates_uniqueness_of :title
5  end
```



Les associations




```
1  class Project < ActiveRecord::Base
2    has_many :tasks
3  end
4
5  class Task < ActiveRecord::Base
6    belongs_to :project
7  end
8
9  mon_project.tasks
10 mon_project.tasks.count
11
12 mon_project.tasks.find(:all, :order => 'title')
13 mon_project.tasks << Task.new(:title => 'foo')
14
15 mon_task.project
```

1 méthode, 1 URL

- Url implicite à partir du nom de contrôleur et du nom de la méthode
- Utilisation de routes nommées
- Définition implicite du fichier de Vue

RESTFULL

- REST (Representational State Transfer)
- Basé sur les 4 verbes HTTP :
  - POST (create)
  - GET (show)
  - PUT (update)
  - DELETE (delete)
- Utilisation des verbes simples avec URL correspondante
- Facilité de création d'une API

Exemple de controller

```
1  class ProjectsController < ApplicationController
2
3    # http://localhost:3000/projects
4    def index
5      @projects = Project.find :all
6    end
7
8    # http://localhost:3000/projects/1
9    def show
10     @project = Project.find params[:id]
11   end
12
13 end
```

Les helpers

- les taglibs de Ruby on Rails
- 1 helper par controller par défaut
- réutilisation des manipulations de vues
- de nombreuses méthodes existantes dans l'API

ActionView

app/views/project/list.html.erb

```
1  <% @project.each do |project| do %>
2    <h1><%= project.title %></h1>
3    <p><%= truncate(project.description, 80) %></p>
4    <p><%= link_to "suite...", :action => "show", :id => project.id %></p>
5  <% end %>
```

app/views/project/show.html.erb

```
1  <h1><%= h @project.title %></h1>
2
3  <p><%= @project.description %></p>
4  <%= render :partial => @project.tasks %>
```

app/views/task/_task.html.erb

```
1  <p><%= task.description %></p>
2  <p><%= task.updated_at %></p>
```

Les tests unitaires

- Dans le dossier /test/units
- Test sur les modéles
- Facilité de créer des jeux de tests
- Base de données indépendante
- Chaque test dans une transaction et ROLLBACK à la fin de chaque test.

Les tests fonctionnels

- Dans le dossier /test/functionals
- Test sur les contrôleurs
- Même système d'injection des jeux de données
- Assertions spécifiques avec vérification du DOM

Les environnements de développement

- Netbeans depuis la version 6.0
- RadRails, plugin d'eclipse
- Ruby In Steel, add-on de Microsoft Visual Studio
- TextMate
- Vim/Emacs

Outils d'aide au développement

- Shell Interactif, irb
- Debugger
- Profiler
- Couverture de code

JRuby implémentation Java de Ruby

- JRuby 1.0 sortie en Juin 2007, 1.1 en Avril 2008
- Entièrement compatible avec Ruby 1.8.6
- Intégration avec toute bibliothèque Java existante

JRuby on Rails

- Depuis JRuby 1.0.2, possibilité de faire tourner Ruby On Rails
- Possibilité de générer un WAR avec le plugin Goldspike
- War généré utilisable dans un serveur d'application

demo