



Let's take the train

Formation Ruby on Rails - Actiane
Samedi 16 Fevrier 2008



1

D'où qu'il vient ?

- États Unis, 2004, un homme : David Heinemeier Hansson
- D'abord pour BaseCamp, 37Signals
- 1er version publique : Juillet 2004
- Dernière version stable : 2.0 - Décembre 2007

2

David Heinemeier Hansson
Né le 15 Octobre 1979 (28 ans)
Danois
Diplômé de l'université de
Copenhague
Collaborateur de 37Signals
(Chicago)



3

Rails is the killer app for Ruby.

Yukihiro Matsumoto

4

Luke, Yoda et les autres.

```
Knight{#id, name, force_level, created_at}

class Knight < ActiveRecord::Base
end

yoda = Knight.new
yoda.name = 'Yoda'
yoda.force_level = 42
yoda.save

luke = Knight.create :name => 'Luke Skywalker',
                    :force_level => 42
```

9

On se croirait sur Aldebaran...

```
knights = Knight.find(:all)

strongest_knight = Knight.find(:all, :order_by => :force_level, :limit => 1)

all_obiwans = Knight.find(:first, :conditions => ['name = ?', 'obiwan'])

palpatine_must_be_destroy = Knight.find(23)
palpatine_must_be_destroy.destroy
```

10

Les migrations.

- Système de version pour la base de donnée.
- Attention, ne sert pas à déployer une application ! Schema.rb sert à ça.
- Faire du SQL sans écrire du SQL.
- Limitation : clé étrangère, trigger, procédure stockée...

11

Mignons vers d'autres cieux.

```
class CreateKnight < ActiveRecord::Migration
  def self.up
    create_table :knights do |t|
      t.column :name, :string, :limit => 50, :null => false
      t.column :force_level, :integer, :null => false
      t.column :create_at, :datetime
    end
  end

  def self.down
    drop_table :knights
  end
end
```

12

Single Table inheritance

- Reporter la notion d'héritage en base
- Une nouvelle table n'est pas DRY
- Utilisation de la même table, avec tous les champs de classes filles
- Assignation de la classe par le champs type

13

Montre l'héritage, papy !

```
Knights(#id, type, name, force_level, created_at)

class Jedi < Knight; end
class Sith < Knight; end

luke = Jedi.find(42)
doku = Sith.find(43)

all_knights = Knight.find(:all)
all_knight.include? luke # true
all_knight.include? doku # true
```

14

Callbacks, ou l'appel de la forêt.

```
(1) before_validation
(2) before_validation_on_create
(-) validate
(-) validate_on_create (4) after_validation_on_create
(3) after_validation (5) before_save
(-) create (6) before_create
(-) after_create (7) after_create
(-) save (8) after_save
```

15

Mise en place du fayot.

```
class Jedi < ActiveRecord::Base
  before_create do
    logger.info("A new Jedi is born and his
name is : ' + self.name.capitalize)
  end
end
```

16

T'as des baskets, tu passe pas.

```
validate
validate_on_create
validate_on_update
validates_acceptance_of
validates_associated
validates_confirmation_of
validates_each
validates_exclusion_of
validates_format_of
validates_inclusion_of
validates_length_of
validates_numericality_of
validates_presence_of
validates_size_of
validates_uniqueness_of
```

17

Relations.

I:I has_one belongs_to_one

I:N belongs_to has_many

N:N has_many_and_belongs_to

N:N has_many ...:through => ...

18

Luke arriva sur Dagoba

```
class Knight < ActiveRecord::Base
  has_one :light_saber
  has_many :padawan, :class_name => :knight
  belongs_to :master, :class_name => :knight
end

class LightSaber
  belongs_to :knight
end
```

19

Tu peux pas test ?

- Test::Unit
- Unitaire pour les modèles.
- Fonctionnel pour les contrôleurs.
- Intégration pour la chaîne complète.

20

Merci !
Des questions ?


