Ruby on Rails

Code d'Armor, 24/02/2014

Bastien Murzeau, PredicSis

Ruby

Ruby

Principes

— Langage de programmation orientée objet

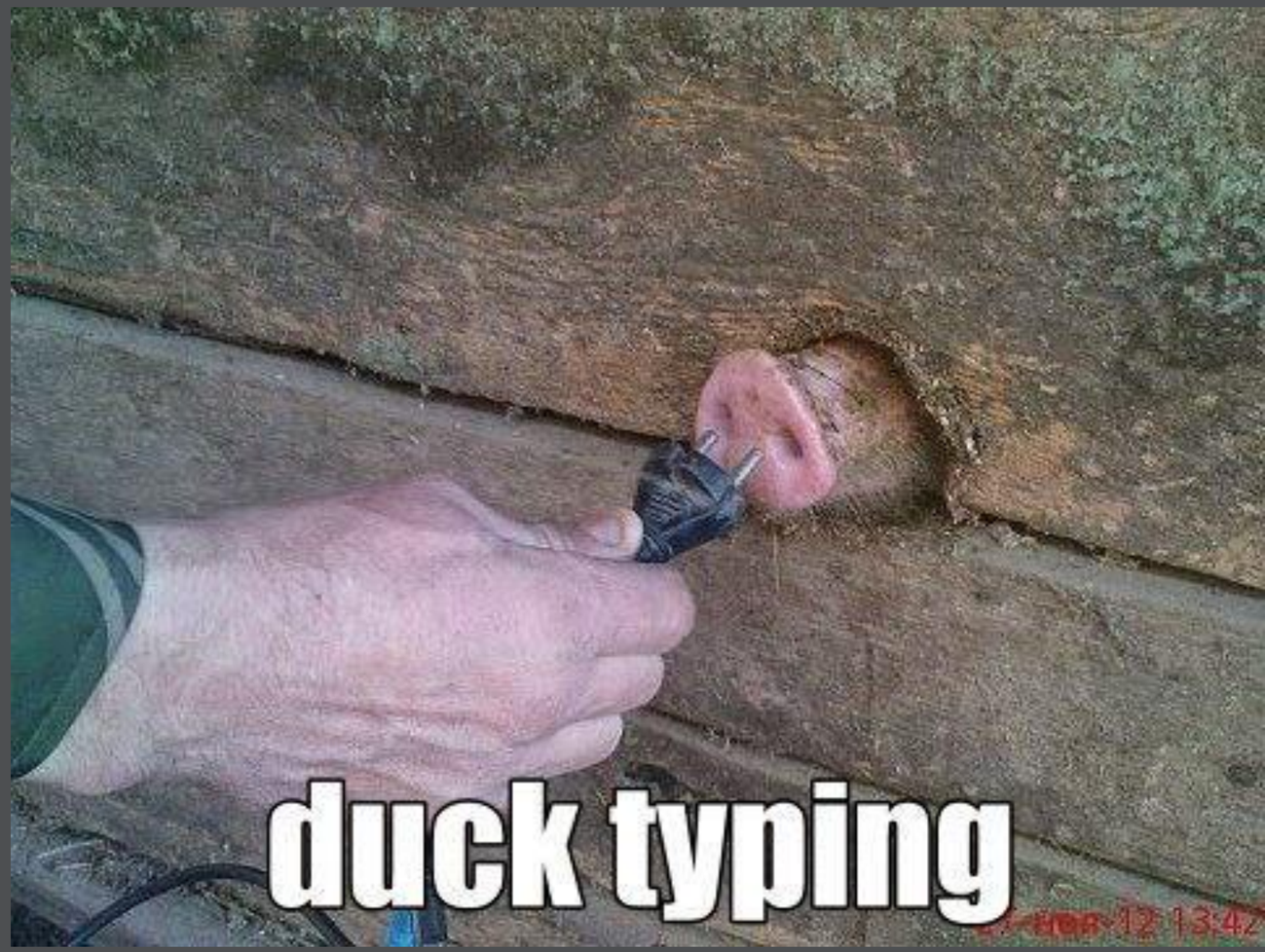
— Open-source & Free ; 1993 (natif MAC OS depuis 2007)

— Dernière génération de langage de scripting comme Python, Perl

Ruby

Principes

- Adapté a l'humain
- Dynamique (*duck typing*)
  - *'If it walks like a duck and talks like a duck, it must be a duck'*
- Léger



Ruby

Pourquoi Ruby

- Programmes plus courts et lisibles
- Effort porté sur la programmation
- Gratuité et possibilité de modifications
- Communauté importante, dynamique et ouverte
- Plus fun !

Ruby

Nommage et méthodes

- En **miniuscule**
  - les variables locales, les argurments de méthodes et les noms de méthodes
- En **majuscule**
  - les noms de classes, modules ou constantes
- Les variables d'instances débutent avec **@**

Ruby

Nommage et méthodes

- Méthodes
  - pas de ';'
  - parenthèses optionnelles
  - '#' pour les commentaires
  - **return** est optionel par défaut la valeur de la dernière expression est retournée

Ruby

Structures de données

- Collections d'objets indexés
- **Array** []
  - la clef est un entier
- **Hash** {}
  - la clef est un objet

Ruby

Classes et modules

- Class
  - no more *getters* and *setters*
  - méthodes public, protected, private
- Module
  - proche d'une Class
  - impossible d'instancier un objet

Ruby

Ruby idiomes

Combinaisons de fonctionnalités ruby

- empty! (bang method): modifie le recepteur
- empty? (predicate method): true ou false
- a || b évalue a
- a ||= b assigne a
- obj = self.new retourne une instance de la classe

Ruby

Installation

- Ruby version 2.2.0
- Gestionnaire de version Rbenv
- Pre-requis : Git
- Console d'éxecution Ruby

```
[~]$ irb
irb(main):001:0> Time.now
=> 2015-02-23 23:04:46 +0100
```

Ruby

Try Ruby in Live!

Code School

Try Ruby

Learn Ruby

Advanced Ruby

*rien besoin d'installer sur votre machine

Rails

Rails

Introduction

- Rails est un framework de développement d’applications web, Ruby est son moteur
- Rails impose le respect de certaines contraintes dans la structuration d’une webapp
  - cela facilite grandement la conception et les développements

Rails

Introduction

— Créer par David Heinemeier Hanson

— 1st release en février 2005

— Rails v4.2.0 en décembre 2014

Rails

Applications connues

Rails

Models, views, controllers

- Modèle MVC pour le développement d’application interactive (1979)
  - Model
  - View
  - Controller

Rails

Models, views, controllers

- Développement de chaque compostant de manière séparé
  - respect des conventions établies évite l’ajout de configurations supplémentaires
  - DRY *Don't Repeat Yourself*

Rails

Models, views, controllers

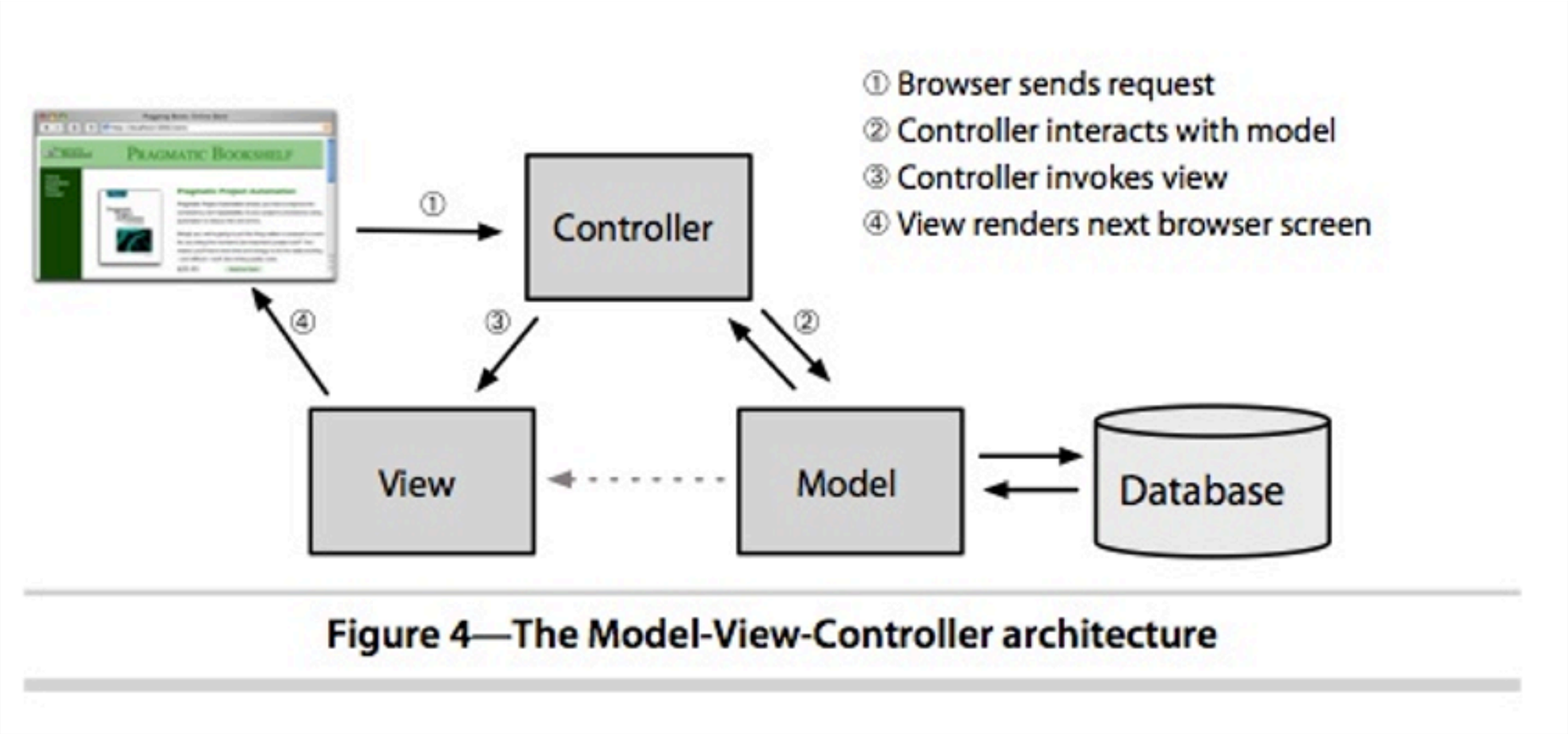


Figure 4—The Model-View-Controller architecture

Rails

Support des modèles, ORM & Active Record

— Les objets sont composés de data et opérations (object-oriented programming)

— Les BDD relationnelles de type SQL sont de simples conteneurs de valeurs

— Rails introduit un mapping entre les données relationnelles et les objets

Rails

Support des modèles, ORM & Active Record

- Object-Relational Mapping (ORM)
  - librairie qui lie les tables de BDD a des des classes
  - pour une table ‘livres’ une classe ‘Livre’
- Ces classes disposent de ‘class-level’ méthodes exécutant des opérations sur les tables

Rails

Support des modèles, ORM & Active Record

- ORM Rails : Active Record
  - nombreux réglages par défaut
  - pas de connexion a la base a gérer
  - effort porté sur la logique métier

Rails

View & controller

- Les composants View et Controller du modèle MVC sont très proches
- **View**
  - construction des réponses affichées
  - inclusion de contenu dynamique, processés par les méthodes du controller ou modèle

Rails

View & controller

- **Controller**
  - pivot de l’application
  - route les requêtes externes vers les actions internes
  - caching, gain de performance
  - gestion des sessions

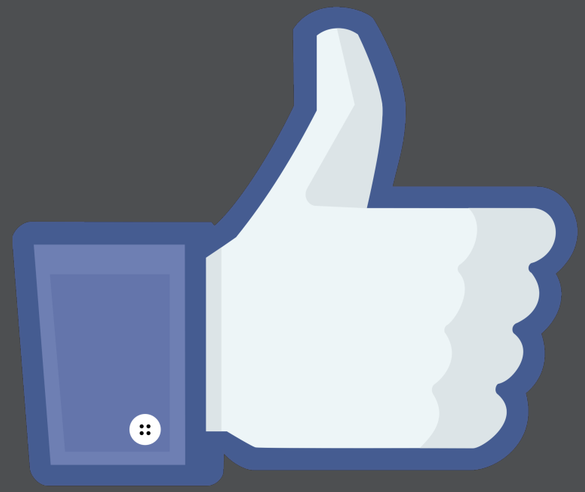
Première application RoR

Nouveau projet

Modélisation d'une ressource

Le modèle MVC en action

REST



Première application RoR

Nouveau projet

- Découverte de la magie de Rails
- Introduction a l'architecture REST
- Gestion de produits

Première application RoR

Nouveau projet

Initialisation du projet Ruby On Rails

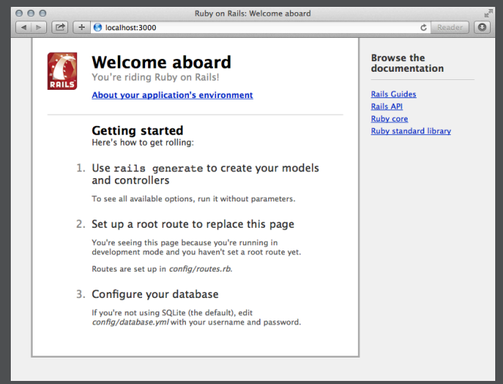
```
[~]$ rails new eshop --database=postgresql
[~]$ cd eshop
[~/eshop]$ git init
[~/eshop]$ git add .
[~/eshop]$ git commit -m 'initial commit'
```

Première application RoR

Nouveau projet

Lancement de l'application

```
[~]$ bundle exec rake db:create
[~]$ bundle exec rails s
```



Première application RoR

Modélisation Product

Un *Product* dispose :

- reference (id)
- name
- price

Première application RoR

Modélisation Product

Création d’une ‘Product resource’, les **products** sont manipulables en tant qu’objets:

— **C**reated

— **R**ead

— **U**pdated

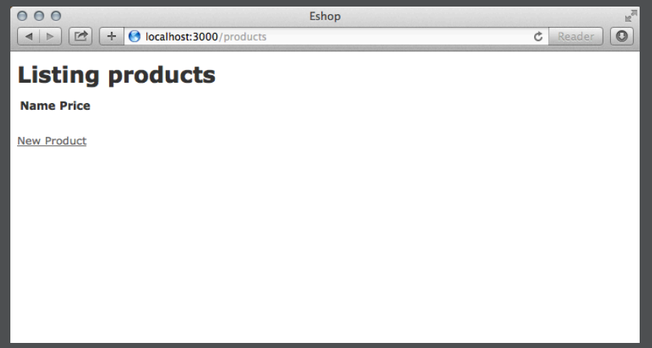
— **D**eleted

Première application RoR

Modélisation Product

Generation automatique avec *scaffold*

```
[~/eshop]$ bundle exec rails generate scaffold Product name:string price:float
[~/eshop]$ bundle exec rake db:migrate
[~/eshop]$ bundle exec rails s
```



Première application RoR

Le modèle MVC en action

8 étapes essentielles :

- Requete **GET** on /products
- Rails route /products sur l’action ‘index’ du **Products controller**
- L’action 'index' demande au model Product tout les products (Product.all)
- Le model Product requete la BDD

Première application RoR

Le modèle MVC en action

- L'ORM retourne la liste des products au controller
- Le controller récupère cette liste qu'il stocke dans la variable @products, qui est ensuite passee la view index
- La view utilise de l'Embedded Ruby' pour generer l'HTML

Première application RoR

REST

REpresentational State Transfer : pure théorie qui modelise les composants d'applications Rails en 'resources'

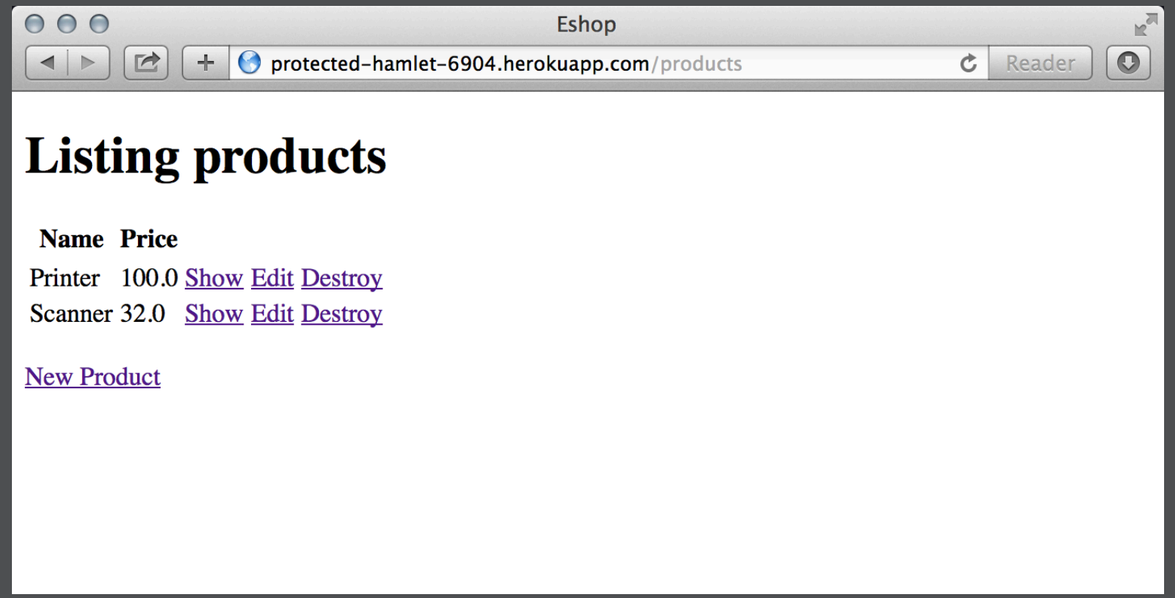
```
[~/eshop]$ bundle exec rake routes
      Prefix Verb   URI Pattern                  Controller#Action
    products GET    /products(.:format)          products#index
             POST   /products(.:format)          products#create
 new_product GET    /products/new(.:format)      products#new
edit_product GET    /products/:id/edit(.:format) products#edit
     product GET    /products/:id(.:format)      products#show
             PATCH  /products/:id(.:format)      products#update
             PUT    /products/:id(.:format)      products#update
             DELETE /products/:id(.:format)      products#destroy
```

Première application RoR

Mise en ligne

Déploiement sur Heroku (PaaS)

```
[~/eshop]$ heroku create --addons heroku-postgresql
[~/eshop]$ git push heroku master
[~/eshop]$ heroku run rake db:migrate
[~/eshop]$ heroku config:set SECRET_KEY_BASE=f...8
[~/eshop]$ heroku open
```



RubyGems

- L'integration de **gems** accelere les developpements
- Les *gems* sont des composants integrables a Rails pour remplir une fonctionnalite precise:
  - Authentification des utilisateurs (Devise)
  - Pagination (Will Paginate)
  - ...
- The Ruby Toolbox ; RubyGems

Merci !

✉ bastien@predicsis.com

@b_a_s_t_i_e_n