Ευάγγελος Σακκόπουλος

```
git commit -m 'deal with it' &&
git push --force origin master
```

1

1

Σύνοψη της σημερινής έννοιας

- Γρήγορη ξενάγηση στο Rails
- ActiveRecord για *διατήρηση* δεδομένων εφαρμογών
- Μοντέλα ActiveRecord ως πόροι και *διαδρομές* προσανατολισμένες σε πόρους για τον χειρισμό τους
- Νέα ξενάγηση στο Rails με συμπερίληψη των εννοιών
- Σύγκριση εννοιών & μηχανισμών του Rails και του Sinatra

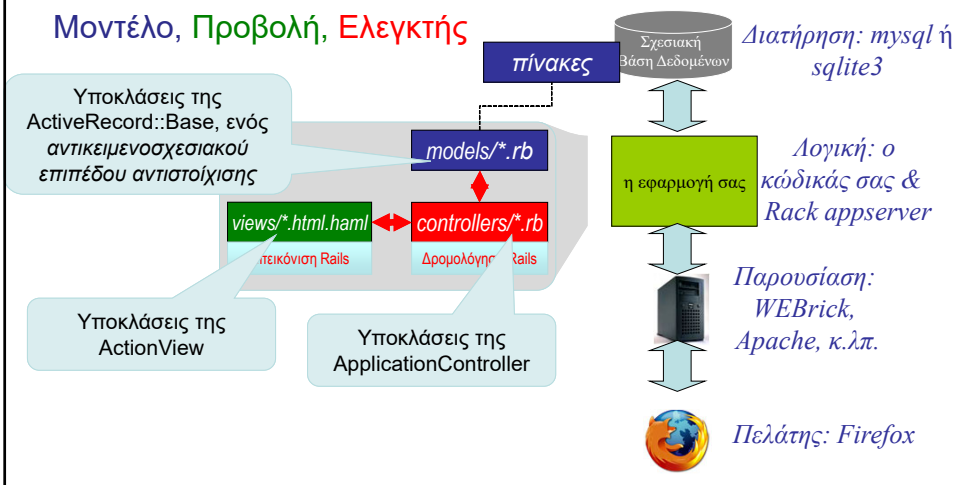
2

2

Καλημέρα Rails :
Από το μηδέν στο CRUD

3

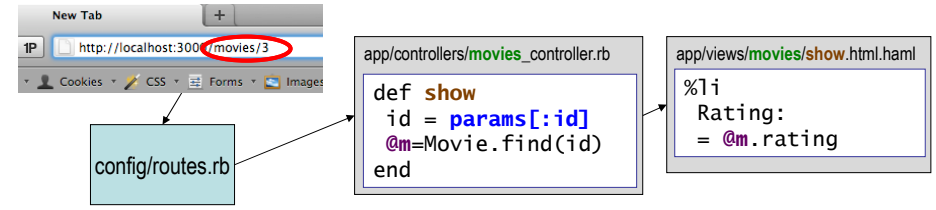
Το Rails ως πλαίσιο εργασίας MVC



4

Περιήγηση σε μια εφαρμογή Rails

1. Οι διαδρομές (στο `routes.rb`) αντιστοιχίζουν εισερχόμενα URL σε *ενέργειες ελεγκτή* και εξάγουν οποιεσδήποτε προαιρετικές *παραμέτρους*
   - Οι παράμετροι-μπαλαντέρ (wildcard) των διαδρομών (π.χ., `:id`), συν οτιδήποτε μετά το «`?`» στο URL, τοποθετούνται στον κατακερματισμό `params[]` που είναι προσπελάσιμος από τις ενέργειες ελεγκτή
2. Οι ενέργειες ελεγκτή ορίζουν *μεταβλητές στιγμιοτύπου,* ορατές στις *προβολές*
   - Οι υποκατάλογοι και τα ονόματα αρχείων στο `views/` ταυτίζονται με ελεγκτές & και ονόματα ενεργειών
3. Η ενέργεια ελεγκτή τελικά *απεικονίζει* (render) μια προβολή

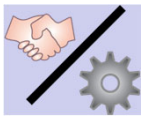



```
get '/movies/:id' => 'movies#show'
```

5

Φιλοσοφία του Rails

- *Σύμβαση αντί διευθέτησης*
  - Αν η ονομασία ακολουθεί συγκεκριμένες συμβάσεις, δεν χρειάζονται αρχεία διευθέτησης



`MoviesController#show` in `movies_controller.rb`
➔ `views/movies/show.html.haml`

- Μην επαναλαμβάνεσαι (Don't Repeat Yourself, DRY)
  - μηχανισμοί για την εξαγωγή κοινής λειτουργικότητας

- Και τα δύο βασίζονται σε χαρακτηριστικά της Ruby:
  - ενδοσκόπηση και μεταπρογραμματισμός
  - μπλοκ (κλειστότητες)
  - υπομονάδες (αναμίξεις)

6

Γιατί κάθε αλληλεπίδραση με μια εφαρμογή Rails πρέπει τελικά να εμφανίζει κάτι;

- ☐ Εξαιτίας της σύμβασης αντί διευθέτησης
- ☐ Επειδή το HTTP είναι ένα πρωτόκολλο αίτησης-απάντησης
- ☐ Επειδή στο MVC υπονοείται ότι κάθε ενέργεια απεικονίζει τη δική της προβολή, και το Rails βασίζεται ιδιαίτερα στο MVC
- ☐ Στην πραγματικότητα, κάποιες αλληλεπιδράσεις με χρήση της AJAX επικοινωνούν με τον διακομιστή *χωρίς* να απεικονίζουν κάτι

7

7

διακοπή

8

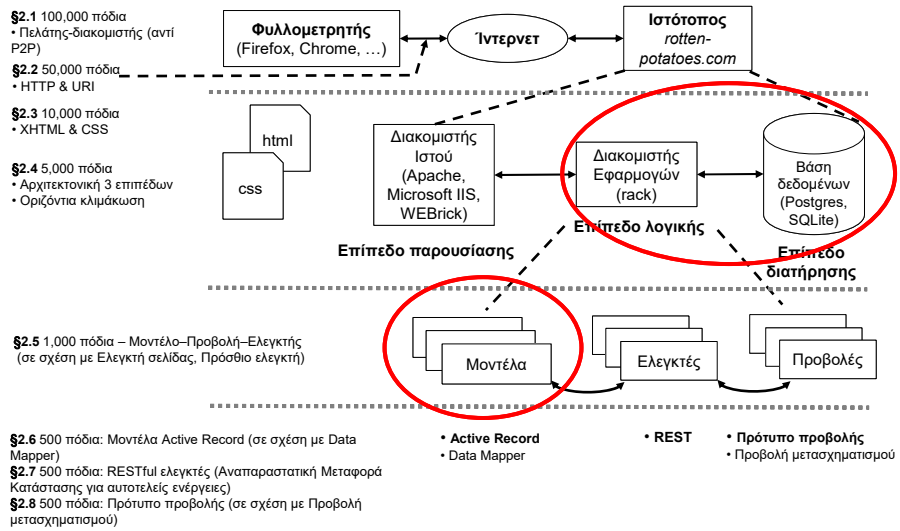
8

Μοντέλα, βάσεις Δεδομένων, και ενεργή εγγραφή

9

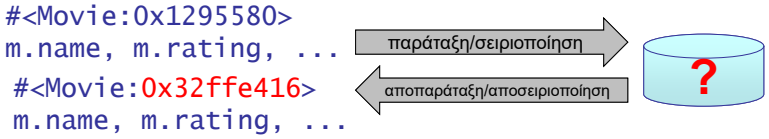
- Πώς θα πρέπει να αποθηκεύουμε και να ανακτούμε *δομημένα δεδομένα με προσανατολισμό στις εγγραφές;*
- Ποια είναι η σχέση μεταξύ της *αποθήκευσης* και του *χειρισμού* δεδομένων *σε* μια *γλώσσα προγραμματισμού;*

10



11

Αντικείμενα στη μνήμη και σε αποθηκευτικό χώρο



- Πώς να αναπαραστήσουμε μόνιμα αντικείμενα σε αποθηκευτικό χώρο
  - Παράδειγμα: `Movie` με ιδιότητες `name` & `rating`
- Βασικές λειτουργίες στο αντικείμενο: CRUD (Create, Read, Update, Delete)
- ActiveRecord: κάθε μοντέλο γνωρίζει πώς να εκτελέσει λειτουργίες CRUD, με κοινούς μηχανισμούς

12

12

Τα μοντέλα του Rails αποθηκεύουν δεδομένα σε σχεσιακές βάσεις δεδομένων (RDBMS)

- Κάθε τύπος μοντέλου λαμβάνει τον δικό του *πίνακα* βάσης δεδομένων
  - Όλες οι γραμμές του πίνακα έχουν την ίδια δομή
  - Μία γραμμή πίνακα == ένα στιγμιότυπο κλάσης μοντέλου
  - Σε κάθε στήλη αποθηκεύεται μια *ιδιότητα* του μοντέλου
  - Κάθε γραμμή έχει μια μοναδική τιμή ως *πρωτεύον κλειδί* (κατά σύμβαση, στο Rails είναι ένας ακέραιος και ονομάζεται *id*)

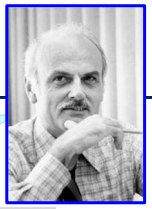
id	rating	title	release_date
2	G	Gone With the Wind	1939-12-15
11	PG	Casablanca	1942-11-26
...	...	...	...
35	PG	Star Wars	1977-05-25

- *Σχήμα:* Συλλογή όλων των πινάκων και της δομής τους

13

CRUD στην SQL

«Ted» Codd



- Η Δομημένη Γλώσσα Ερωτημάτων (SQL) είναι η γλώσσα ερωτημάτων στα RDBMS
- Το Rails *παράγει* εντολές SQL κατά τον χρόνο εκτέλεσης, με βάση τον κώδικά σας Ruby
- 4 βασικές λειτουργίες σε μια γραμμή πίνακα: Δημιουργία (**C**reate), Ανάγνωση (**R**ead), Ενημέρωση (**U**pdate) ιδιοτήτων, Διαγραφή (**D**elete)

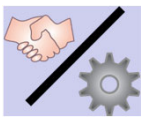

```
INSERT INTO users (username, email, birthdate)
    VALUES ("fox", "fox@cs.berkeley.edu", "1968-05-12"),
           ("patterson", "pattrsn@cs.berkeley.edu", "????")
SELECT *
  FROM users
  WHERE (birthdate BETWEEN "1987-01-01" AND “2000-01-01”)
UPDATE users
  SET email = "armandofox@gmail.com"
  WHERE username="fox"
DELETE FROM users WHERE id=1
```

14

Η πτυχή Ruby ενός μοντέλου

- Υποκλάση από την `ActiveRecord::Base`
  - «συνδέει» ένα μοντέλο στη βάση δεδομένων
  - παρέχει λειτουργίες CRUD στο μοντέλο

http://pastebin.com/ruu5y0D8



- Το όνομα πίνακα βάσης δεδομένων παράγεται από το όνομα του μοντέλου: `Movie`➔`movies`
- Τα ονόματα στηλών του πίνακα βάσης δεδομένων είναι συναρτήσεις getter & setter για τις ιδιότητες του μοντέλου
- *Παρατήρηση: οι συναρτήσεις getter και setter δεν τροποποιούν απλώς τις μεταβλητές στιγμιοτύπου!*

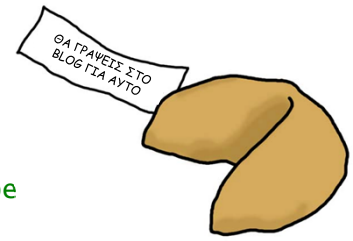
15

Create: new ≠ save

- Πρέπει να καλείται η `save` ή `save!` σε ένα στιγμιότυπο μοντέλου ActiveRecord για την αποθήκευση των αλλαγών στη βάση δεδομένων
  - Η έκδοση '!' καταθέτει εξαίρεση αν αποτύχει η λειτουργία
  - η `create` απλώς συνδυάζει τις `new` και `save`
- Αφού δημιουργηθεί το αντικείμενο, λαμβάνει ένα πρωτεύον κλειδί (στήλη `id` σε κάθε πίνακα μοντέλου ActiveRecord)
  - αν το `x.id` είναι `nil` ή το `x.new_record?` είναι αληθές, το `x` δεν αποθηκεύτηκε ποτέ
  - Αυτές οι συμπεριφορές κληρονομούνται από την `ActiveRecord::Base`—δεν ισχύει για τα αντικείμενα της Ruby γενικά

16

Υποθέστε ότι ο πίνακας `fortune_cookies`
έχει μια στήλη `fortune_text`
Ποια από τις μεθόδους στιγμιοτύπου της
`FortuneCookie < ActiveRecord::Base`
ΔΕΝ θα επιστρέψει ένα silly_fortune (αν επιστρέψει κάτι);



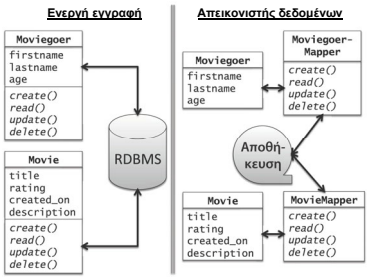
- ☐
```
def silly_fortune_1
  @fortune_text + 'in bed'
end
```
- ☐
```
def silly_fortune_2
  self.fortune_text + 'in be
end
```
- ☐
```
def silly_fortune_3
  fortune_text + 'in bed'
end
```
- ☐ Όλες θα επιστρέψουν ένα silly_fortune

17

17

Εναλλακτική: DataMapper

- Το Data Mapper συσχετίζει ένα ξεχωριστό *mapper* («αντιστοίχιση») με κάθε μοντέλο
  - Ιδέα: διατήρηση της αντιστοίχισης *ανεξάρτητης* από τον συγκεκριμένο αποθηκευτικό χώρο δεδομένων => δουλεύει με τους περισσότερους τύπους βάσεων δεδομένων
  - Χρησιμοποιείται από την Google AppEngine
  - Αρνητικό: δεν εκμεταλλεύεται χαρακτηριστικά RDBMS για να απλοποιεί σύνθετα ερωτήματα & σχέσεις
- Θα το δούμε πάλι όταν μιλήσουμε για *συσχετίσεις*



18

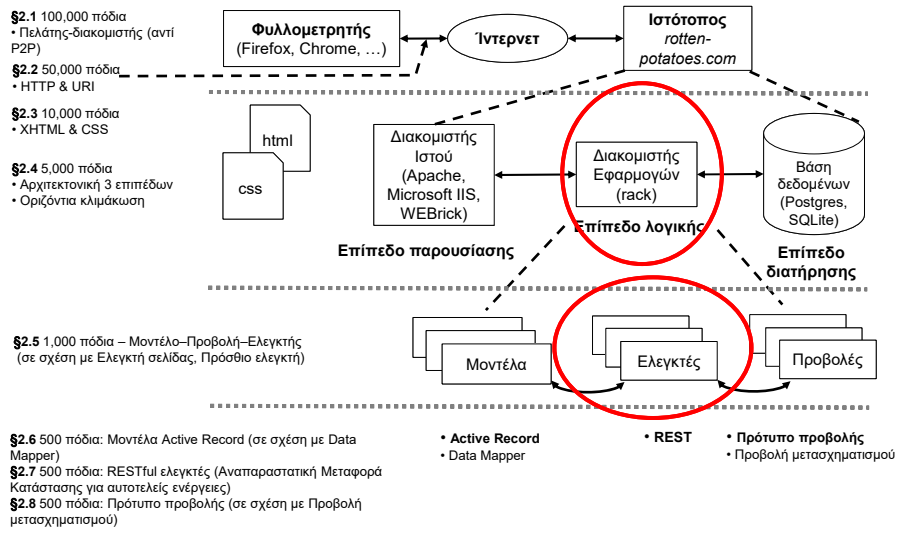
18

τέλος

19

RESTful διαδρομές πόρων στο Rails

20



21

Βασικά του `config/routes.rb`

```
get '/pastries/:flavor'=> 'pastries#eat',
  :as => 'eat_dessert'
```

- Διαδρομές που ταιριάζουν με τη διαδρομή στο `PastriesController#eat`
- Συμπληρώνει το `params[:flavor]` από το URL
- Σε μια προβολή:

```
link_to 'Eat', eat_dessert_path('cherry')
```

➔ `<a href="/pastries/cherry">Eat</a>`

- *Ιδέα DRY: η λογική διαδρομής εκτελείται και με τους δύο τρόπους*
- Γρήγορη αναφορά:

`http://guides.rubyonrails.org/routing.html`

22

22

CRUD σε RESTful πόρο: resources :movies

```
get '/movies' => 'movies#index', :as => 'movies'
get '/movies/:id/new'=>'movies#new', :as=>'new_movie'
post '/movies' => 'movies#create', :as => 'movie'
get '/movies/:id' => 'movies#show', :as => 'movie'
get '/movies/:id/edit' => 'movies#edit', :as => 'edit_movie'
put '/movies/:id'=>'movies#update', :as=>'movie'
delete '/movies/:id' => 'movies#destroy', :as=>'movie'
```

I
C
R
U
D

Διαδρομή	Ενέργεια
GET /movies/3	Εμφάνιση πληροφοριών για την ταινία με ID=3
POST /movies	Δημιουργία νέας ταινίας από τα δεδομένα της φόρμας
PUT /movies/5	Ενημέρωση της ταινίας με ID 5 με τα δεδομένα της φόρμας
DELETE /movies/5	Διαγραφή της ταινίας με ID=5

23

23

CRUD σε RESTful πόρο: resources :movies

```
get '/movies' => 'movies#index', :as => 'movies'
get '/movies/:id/new'=>'movies#new', :as=>'new_movie'
post '/movies' => 'movies#create', :as => 'movie'
get '/movies/:id' => 'movies#show', :as => 'movie'
get '/movies/:id/edit' => 'movies#edit', :as => 'edit_movie'
put '/movies/:id'=>'movies#update', :as=>'movie'
delete '/movies/:id' => 'movies#destroy', :as=>'movie'
```

```
form_for movie_path(5)
  <form action="/movies/5" method="post">
form_for movie_path(5), :method => :put
  <form action="/movies/5" method="post">
    <input type="hidden" name="_method" value="put"> ...
link_to 'Edit', edit_movie_path(5)
  <a href="/movies/5/edit">Edit</a>
link_to 'List All', movies_path
  <a href="/movies">List All</a>
```

24

24

Ποια πρόταση **ΔΕΝ** είναι αληθής σχετικά με τις RESTful διαδρομές του Rails και τους πόρους στους οποίους αναφέρονται:

- ☐ Ένας *πόρος* μπορεί να είναι υπάρχον περιεχόμενο ή μια αίτηση για να τροποποιηθεί κάτι.
- ☐ Σε μια εφαρμογή MVC, κάθε διαδρομή πρέπει τελικά να πυροδοτήσει μια ενέργεια ελεγκτή.
- ☐ Ένα συνηθισμένο σύνολο RESTful ενεργειών είναι οι ενέργειες CRUD σε μοντέλα.
- ☐ Η διαδρομή πάντα περιέχει μία ή περισσότερες παραμέτρους-μπαλαντέρ, όπως το `:id`, για τον προσδιορισμό ενός συγκεκριμένου στιγμιοτύπου πόρου στη λειτουργία

25

25

τέλος

26

26

Βάσεις δεδομένων & μεταβάσεις

27

Τα δεδομένα των πελατών είναι χρυσάφι!

- Πώς αποφεύγουμε την αλλοίωσή τους όταν πειραματιζόμαστε ή αναπτύσσουμε νέα χαρακτηριστικά;
- Πώς παρακολουθούμε και διαχειριζόμαστε τις *αλλαγές σχήματος*;
- …η απάντηση και στα δύο είναι *αυτοματισμός!*

28

28

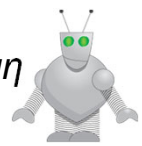
Πολλά περιβάλλοντα, πολλές βάσεις δεδομένων

- Λύση στο Rails: καθένα από τα *περιβάλλοντα* ανάπτυξης, παραγωγής και ελέγχου έχει τη δική του βάση δεδομένων (ΒΔ)
  - Διαφορετικοί τύποι ΒΔ κατάλληλοι για το καθένα!
- Πώς γίνονται *αλλαγές* στη ΒΔ, αφού θα πρέπει να επαναλαμβάνονται οι αλλαγές στη ΒΔ παραγωγής;
- Λύση στο Rails: *μετάβαση* (migration)—ένα σενάριο (script) που περιγράφει τις αλλαγές και μεταφέρεται μεταξύ διαφορετικών τύπων ΒΔ

29

Πλεονεκτήματα μετάβασης

- Μπορείτε να προσδιορίζετε κάθε μετάβαση και να γνωρίζετε ποιες εφαρμόζονται και πότε
  - Πολλές μεταβάσεις μπορούν να δημιουργούνται έτσι ώστε να είναι *αντιστρέψιμες*
- Δυνατότητα διαχείρισης με έλεγχο εκδόσεων
- *Αυτοματοποιημένη == αξιόπιστα επαναλήψιμη*
- Θέμα: *μην το κάνετε – αυτοματοποιήστε το*
  - *καθορίστε* τι να κάνετε, δημιουργήστε εργαλεία για αυτοματοποίηση



30

Συναντήστε μια γεννήτρια κώδικα

`rails generate migration CreateMovies`

http://pastebin.com/VYwbc5fq

- Σημείωση: αυτό απλώς *δημιουργεί* τη μετάβαση. Δεν την έχουμε *εφαρμόσει*.
- Εφαρμογή της μετάβασης στην ανάπτυξη: `rake db:migrate`
- Εφαρμογή της μετάβασης στην παραγωγή: `heroku rake db:migrate`
- Η εφαρμογή της μετάβασης επιπλέον καταγράφει στην ίδια τη ΒΔ ποιες μεταβάσεις έχουν εφαρμοστεί

31

Μαγειρική στο Rails #1

- Αύξηση της λειτουργικότητας της εφαρμογής == προσθήκη μοντέλων, προβολών, ενεργειών ελεγκτή

Για να *προσθέσετε ένα νέο μοντέλο* σε μια εφαρμογή Rails :

- (ή να αλλάξετε/προσθέσετε ιδιότητες σε ένα υπάρχον μοντέλο)

1. Δημιουργήστε μια μετάβαση που περιγράφει τις αλλαγές: `rails generate migration` (παρέχει στερεότυπο κώδικα)
2. Εφαρμόστε τη μετάβαση: `rake db:migrate`
3. Αν είναι νέο μοντέλο, δημιουργήστε το αρχείο μοντέλου `app/models/`*μοντέλο*`.rb`
4. Ενημερώστε το σχήμα ΒΔ ελέγχου: `rake db:test:prepare`
5. Τελικά εγκαταστήστε: `heroku rake db:migrate`

32

Με βάση αυτά που έχετε δει στο Rails, τι είδος αντικειμένου είναι *πιθανό* να παραχθεί στον κώδικα μετανάστευσης:

```
def up
  create_table 'movies' do |t|
    t.datetime 'release_date' ...
  end
end
```

- ☐ Ένα αντικείμενο που αναπαριστά μια βάση δεδομένων
- ☐ Ένα αντικείμενο που αναπαριστά ένα στιγμιότυπο ενός μοντέλου
- ☐ Ένα αντικείμενο που αναπαριστά έναν πίνακα
- ☐ θα μπορούσε να είναι οτιδήποτε!!!

33

33

34

34

Μοντέλα: Εύρεση, ενημέρωση, διαγραφή

35

Read: εύρεση στοιχείων στη ΒΔ

- η μέθοδος κλάσης `where` επιλέγει αντικείμενα με βάση τις ιδιότητες· επιστρέφει *ένα enumerable*

```
Movie.where("rating='PG'")
Movie.where('release_date < :cutoff and
     rating = :rating',
     :rating => 'PG', :cutoff => 1.year.ago)
Movie.where("rating=#{rating}") # ΚΑΚΗ ΙΔΕΑ!
Movie.find(3) #εξαίρεση αν δε βρεθεί
```

- Μπορούν να συνδεθούν διαδοχικά με αποδοτικό τρόπο

```
kiddie = Movie.where("rating='G'")

old_kids_films =
  kiddie.where("release_date < ?",30.years.ago)
```

36

Update: 2 τρόποι

- Έστω `m=Movie.where(title: 'The Help')`
- Τροποποιήστε ιδιότητες, μετά αποθηκεύστε το αντικείμενο
```
m.release_date='2011-Aug-10'
m.save!
```
- Ενημερώστε τις ιδιότητες σε ένα υπάρχον αντικείμενο
```
m.update_attributes(
   release_date: '2011-Aug-10')
```
- Εκτελείται ως συναλλαγή (transaction): είτε ενημερώνονται όλες οι ιδιότητες είτε δεν ενημερώνεται καμία

37

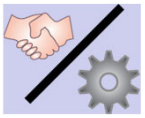
Η διαγραφή με Delete είναι απλή

- Σημείωση! Η destroy είναι μέθοδος *στιγμιοτύπου*
```
m = Movie.where(name: 'The Help')
m.destroy
```
- Υπάρχει και η `delete`, η οποία δεν πυροδοτεί *αντίστροφες κλήσεις κύκλου ζωής* που θα συζητήσουμε αργότερα (έτσι, αποφύγετέ την)
- Όταν καταστραφεί ένα αντικείμενο ActiveRecord μπορείτε να προσπελάσετε *αλλά όχι να τροποποιήσετε* ένα αντικείμενο στη μνήμη
```
m.title = 'Help'  # ΑΠΟΤΥΓΧΑΝΕΙ
```
…με την ευκαιρία, πώς υλοποιείται αυτό;

38

Περίληψη: Εισαγωγή στο ActiveRecord

- Η υποκλάση από την `ActiveRecord::Base` «συνδέει» ένα μοντέλο στη βάση δεδομένων
  - **C** (`save`/`create`), **R** (`where`, `find`), **U** (`update_attributes`), **D** (`destroy`)
- Με τη σύμβαση αντί διευθέτησης αντιστοιχίζονται:
  - όνομα μοντέλου σε όνομα πίνακα ΒΔ
  - Συναρτήσεις getter/setter σε στήλες πίνακα ΒΔ
- Αντικείμενο στη μνήμη ≠ γραμμή στη ΒΔ!
  - Πρέπει να χρησιμοποιείται η `save` για διατήρηση
  - Η `destroy` δεν καταστρέφει το αντίγραφο μνήμης



39

Υποθέστε ότι έχουμε κάνει το εξής:
`movie = Movie.where(:title => 'Amelie')`
Μετά *ένας άλλος χρήστης της εφαρμογής* αλλάζει την καταλληλότητα της ταινίας, *και ενημερώνει τη βάση δεδομένων*. Την αμέσως επόμενη στιγμή, η τιμή της `movie`:

- ☐ θα ενημερωθεί αυτόματα επειδή ένα μοντέλο ActiveRecord έχει «συνδέσει» την εφαρμογή σας στη βάση δεδομένων
- ☐ θα ενημερωθεί αυτόματα εξαιτίας της χρήσης μεταπρογραμματισμού από το ActiveRecord
- ☐ *δεν* θα ενημερωθεί αυτόματα, αλλά μπορεί να ενημερωθεί «χειρονακτικά» με επανεκτέλεση της `movie = Movie.where("title='Amelie'")`
- ☐ μπορεί να είναι ακαθόριστη ή να εξαρτάται από την υλοποίηση

40

40

41

Ελεγκτές & προβολές

42

Μαγειρική στο Rails #2

- Για να *προσθέσετε μια νέα ενέργεια* σε μια εφαρμογή Rails

1. Δημιουργήστε *διαδρομή* στο `config/routes.rb` αν χρειάζεται
2. Προσθέστε την *ενέργεια* (μέθοδο) στο κατάλληλο `app/controllers/*_controller.rb`
3. Διασφαλίστε ότι υπάρχει κάτι για *απεικόνιση* από την ενέργεια στο `app/views/model/action.html.haml`

- Θα χρησιμοποιήσουμε την ενέργεια & προβολή Show (το βιβλίο εξηγεί την ενέργεια & προβολή Index)

43

Αρμοδιότητες MVC

- *Μοντέλο:* μέθοδοι για τη λήψη/χειρισμό δεδομένων
  `Movie.where(...), Movie.find(...)`
- *Ελεγκτής:* λήψη δεδομένων από το μοντέλο, διάθεσή τους στην Προβολή

```
def show
  @movie = Movie.find(params[:id])
end
```

Οι μεταβλητές στιγμιοτύπου που ορίζονται στον ελεγκτή γίνονται διαθέσιμες στην προβολή

Αν δεν υπάρχουν άλλες πληροφορίες, το Rails θα αναζητήσει το app/views/movies/show.html.haml

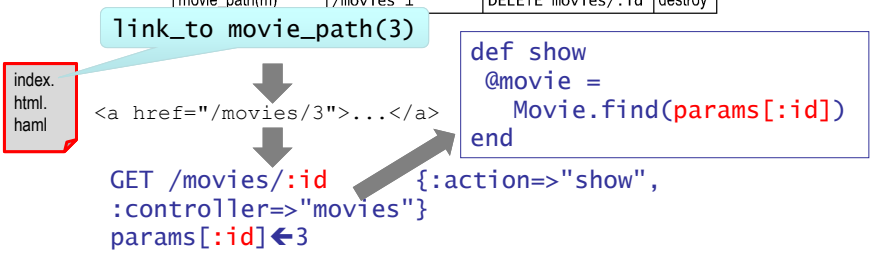
- *Προβολή:* εμφανίζει δεδομένα, επιτρέπει την αλληλεπίδραση με τον χρήστη
  - Δείχνει λεπτομέρειες μιας ταινίας (περιγραφή, καταλληλότητα)
- Αλλά…
  - Τι άλλο μπορεί να κάνει ο χρήστης από αυτή τη σελίδα;
  - Πώς ο χρήστης φτάνει σε αυτή τη σελίδα;

http://pastebin.com/kZCB3uNj

44

Πώς φτάσαμε εδώ: βοηθητικές συναρτήσεις URI

Βοηθητική μέθοδος	Επιστρεφόμενο URI	Διαδρομή και ενέργεια RESTful	
movies_path	/movies	GET /movies	index
movies_path	/movies	POST /movies	create
new_movie_path	/movies/new	GET /movies/new	new
edit_movie_path(m)	/movies/1/edit	GET /movies/:id/edit	edit
movie_path(m)	/movies/1	GET /movies/:id	show
movie_path(m)	/movies/1	PUT /movies/:id	update
movie_path(m)	/movies 1	DELETE movies/:id	destroy



45

Τι άλλο μπορούμε να κάνουμε;

- Να επιτρέψουμε το χρήστη να επιστρέψει στη λίστα ταινιών;
- RESTful βοηθός URI για τη διάσωση ξανά:
  - `movies_path` χωρίς ορίσματα διασυνδέει στην ενέργεια Ευρετηρίου

```
=link_to 'Back to List', movies_path
```

Βοηθητική μέθοδος	Επιστρεφόμενο URI	Διαδρομή και ενέργεια RESTful	
movies_path	/movies	GET /movies	index
movies_path	/movies	POST /movies	create
new_movie_path	/movies/new	GET /movies/new	new
edit_movie_path(m)	/movies/1/edit	GET /movies/:id/edit	edit
movie_path(m)	/movies/1	GET /movies/:id	show
movie_path(m)	/movies/1	PUT /movies/:id	update
movie_path(m)	/movies 1	DELETE movies/:id	destroy

ESaaS, Εικ. 4.7

46

Ποιες προτάσεις είναι ΣΩΣΤΕΣ:

α) Μια διαδρομή αποτελείται από ένα URI **και** μια μέθοδο HTTP

β) Μια διαδρομή **πρέπει** να παράγεται από βοηθούς URI του Rails

γ) Μια διαδρομή **μπορεί** να παράγεται από βοηθούς URI του Rails

- ☐ Μόνο το (α) είναι σωστό
- ☐ Μόνο το (γ) είναι σωστό
- ☐ Μόνο τα (α) και (β) είναι σωστά
- ☐ Μόνο τα (α) και (γ) είναι σωστά

47

47

τέλος

48

48

Όταν τα πράγματα πάνε λάθος: Αποσφαλμάτωση

49

Η αποσφαλμάτωση του SaaS μπορεί να είναι δύσκολη

- Σφάλματα που προκύπτουν νωρίς στη ροή μπορεί να εμφανιστούν πολύ αργότερα
  URI➔διαδρομή➔ελεγκτής➔μοντέλο➔προβολή➔απεικόνιση
- Ο εντοπισμός ή η αναπαραγωγή του σφάλματος μπορεί να είναι δύσκολα αν το σφάλμα επηρεάζει μόνο μερικούς χρήστες, διαδρομές, κ.λπ.

Τι	Ανάπτυξη;	Παραγωγή;
“printf debugging”	✓	
`rails console`	✓	
Καταγραφή (Logging)	✓	✓
`rails server --debugger`	✓	

50

RASP

- ***Η αποσφαλμάτωση είναι γεγονός.***
- Διάβασε (**R**ead) το μήνυμα σφάλματος. Πραγματικά διάβασέ το.
- Ρώτα (**A**sk) έναν συνάδελφο μια *εμπεριστατωμένη* ερώτηση.
- Αναζήτησε (**S**earch) χρησιμοποιώντας το StackOverflow, μια μηχανή αναζήτησης, κ.λπ.
  - Ειδικά για σφάλματα που αφορούν συγκεκριμένες εκδόσεις βιβλιοθηκών gem, λειτουργικού συστήματος, κ.λπ.
- Δημοσίευσε (**P**ost) στο StackOverflow, χώρους συζητήσεων της τάξης, κ.λπ.
  - Οι άλλοι είναι απασχολημένοι όπως και εσύ. Βοήθησέ τους παρέχοντας *ελάχιστες αλλά τεκμηριωμένες* πληροφορίες.

51

Ανάγνωση μηνυμάτων σφαλμάτων στη Ruby

- Το *ίχνος κλήσεων* (backtrace) σας δείχνει τη στοίβα κλήσης (τη διαδρομή) στο σημείο διακοπής.
- Ένα πολύ συχνό μήνυμα:
  `undefined method 'foo' for nil:NilClass`
- Συχνά, σημαίνει ότι μια ανάθεση απέτυχε σιωπηρά και δεν κάνατε έλεγχο σφαλμάτων:

```
@m = Movie.find_by_id(id)  # πιθανό nil
@m.title   # θα αποτύχει: 'undefined method'
```

52

Ενορχήστρωση (δηλαδή «εκτύπωση των τιμών των στοιχείων»)

- Στις προβολές:

```
= debug(@movie)
= @movie.inspect
```

- Στο αρχείο καταγραφής, συνήθως από τη μέθοδο ελεγκτή:

```
logger.debug(@movie.inspect)
```

- Μην χρησιμοποιείτε μόνο την `puts` ή `printf`! Αυτό δεν βγάζει πουθενά στην παραγωγή.

53

Αναζήτηση: Χρησιμοποιήστε το Διαδίκτυο για να απαντήσετε ερωτήσεις

- Ψάξτε το στη Google
  - "How do I format a date in Ruby?"
  - "How do I add Rails routes beyond CRUD?"
- Ψάξτε στην τεκμηρίωση
  - api.rubyonrails.org, πλήρης τεκμηρίωση του Rails με δυνατότητα αναζήτησης (βεβαιωθείτε για τη σωστή έκδοση του Rails!)
  - ruby-doc.org, πλήρης τεκμηρίωση της Ruby με δυνατότητα αναζήτησης (περιλαμβάνει και βοήθεια για τις καθιερωμένες βιβλιοθήκες)
- Ψάξτε στο StackOverflow

54

1. DRY & RASP: Επαναχρησιμοποιήστε τη δουλειά άλλων

- «Με λίγη αναζήτηση για gem/βιβλιοθήκες, θα μπορούσαμε να δουλέψουμε σε νέα χαρακτηριστικά αντί να ανακαλύπτουμε ξανά τον τροχό»
- «Η εκμάθηση της αποτελεσματικής χρήσης της Google και του StackOverflow μας έκανε πραγματικά παραγωγικούς»
- «Για κάθε 5 λεπτά που ξόδευα στη συγγραφή κώδικα, ξόδευα μία ώρα για αναζήτηση στο Διαδίκτυο»
- «Εύχομαι να είχα ψάξει/ζητήσει βοήθεια νωρίτερα, αντί να χτυπώ το κεφάλι μου στον τοίχο μόνος.»
- «Δεν είχα συνειδητοποιήσει πόση ώρα απαιτούνταν για να δημιουργήσω κάτι από το μηδέν χωρίς κάποιον αρχικό κώδικα και χωρίς την καθοδήγηση κάποιου καθηγητή.»

55

Άλλες παρατηρήσεις

- RASP
  - για λάθη/σφάλματα: δώστε μια αφήγηση του τι έχετε κάνει μέχρι εκείνη τη στιγμή (όχι ένα ίχνος στοίβας, και όχι ένα λιτό μήνυμα χωρίς όνομα αρχείου ή αριθμό γραμμής)
  - για αναδόμηση: Τι ακριβώς θέλετε να πετύχετε με την αναδόμηση; Τι έχετε ήδη σκεφτεί να κάνετε;
- RTFB ή WTFV (διάβασε το * βιβλίο ή παρακολούθησε το * βίντεο)

56

τέλος

57

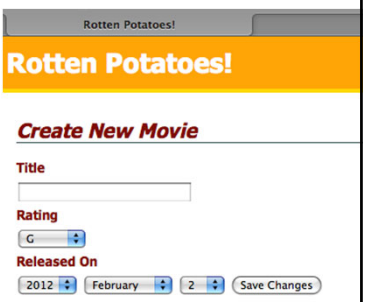
Φόρμες

(Τεχνολογία ανάπτυξης λογισμικού ως υπηρεσίας §4.6)

58

Χρήση φορμών

- Η δημιουργία ενός πόρου συνήθως χρειάζεται 2 αλληλεπιδράσεις
  - `new:` Ανάκτηση κενής φόρμας
  - `create:` Υποβολή συμπληρωμένης φόρμας
- Πώς παράγεται/εμφανίζεται;
- Πώς λαμβάνονται τιμές που συμπληρώνονται από τον χρήστη;
- Τι πρέπει να «επιστραφεί» (απεικονιστεί);



59

Μαγειρική στο Rails #3

- Για να δημιουργήσετε μια καινούρια φόρμα προς υποβολή:

1. Προσδιορίστε την ενέργεια που εξυπηρετεί την ίδια τη φόρμα
2. Προσδιορίστε την ενέργεια που λαμβάνει την *υποβολή*
3. Δημιουργήστε διαδρομές, ενέργειες, προβολές για την καθεμία

- Οι ιδιότητες `name` των στοιχείων της φόρμας θα εμφανιστούν ως κλειδιά στο `params[]`
- Παρέχονται βοηθητικές μέθοδοι για πολλά συνηθισμένα στοιχεία

60

Δημιουργία της φόρμας

- Ανατομία μιας φόρμας σε HTML http://pastebin.com/k8Y49EhE
  - οι ιδιότητες *action* και *method* (δηλαδή, η διαδρομή)
  - μόνο *τα επώνυμα στοιχεία εισόδου* της φόρμας θα υποβληθούν
- Δημιουργία της φόρμας στο Rails
  - συχνά μπορείτε να χρησιμοποιείτε μια βοηθητική μέθοδο URI για την ιδιότητα *action*, αφού είναι μόνο το τμήμα URI μιας διαδρομής (εξακολουθείτε να χρειάζεστε τη *method*)
  - οι *βοηθητικές μέθοδοι πεδίων φόρμας* (δείτε api.rubyonrails.org) παράγουν στοιχεία εισόδου της φόρμας με βολικά ονόματα http://pastebin.com/3dGWsSq8

61

Ποιο από αυτά θα ήταν έγκυρο για τη δημιουργία της φόρμας που, όταν υποβάλλεται, θα καλούσε την ενέργεια δημιουργίας νέας ταινίας (Create New Movie);

- ☐ `= form_tag movies_path do ... end`
- ☐ `%form{:action => movies_path, :method => :post}`
- ☐ `%form{:action => '/movies', :method => 'post'}`
- ☐ Όλα τα παραπάνω

62

62

τέλος

63

63

Μικρογρίφος

goo.gl/NqkrBW

bit.ly/CS169-quiz4

64

65

Ανακατεύθυνση, flash και session

(Τεχνολογία ανάπτυξης λογισμικού ως υπηρεσίας §4.7)

66

Λήψη της φόρμας

- Ένα κομψό κόλπο: χρησιμοποιήστε τον αποσφαλματωτή για να επιθεωρήσετε τι συμβαίνει
  - ξεκινήστε με `rails server --debugger`
  - εισάγετε το `debugger` εκεί όπου θέλετε να σταματήσετε
  - λεπτομέρειες & σύνοψη των εντολών: ESaaS §4.7
- Προσοχή: το `params[:movie]` *είναι ένας κατακερματισμός*, λόγω του τρόπου με τον οποίο ονομάσαμε τα πεδία φόρμας
  - Αρκετά βολικό, αφού αυτό χρειάζεται η `Movie.create!`

67

Ποια προβολή πρέπει να απεικονιστεί για την ενέργεια create;

- Ιδιωματισμός: *ανακατευθύνετε* τον χρήστη σε μια πιο χρήσιμη σελίδα.
  - π.χ., λίστα ταινιών, αν είναι επιτυχής η create
  - π.χ., φόρμα New Movie, αν είναι ανεπιτυχής
- Η ανακατεύθυνση πυροδοτεί μια *εντελώς καινούρια αίτηση HTTP*
  - Πώς να πληροφορήσετε τον χρήστη *γιατί* έγινε η ανακατεύθυνση;
- Λύση: `flash[]`— κραυγάζει σαν κατακερματισμός που *διατηρείται μέχρι το τέλος της **επόμενης** αίτησης*
  - `flash[:notice]` συμβατικά για πληροφορίες
  - `flash[:warning]` συμβατικά για «λάθη»

68

Flash & Session

- `session[]`: σαν κατακερματισμός που διατηρείται για πάντα
  - `reset_session` προκαλεί πλήρη εκκαθάριση
  - `session.delete(:some_key)`, σαν κατακερματισμός
- Εξ ορισμού, στα cookies αποθηκεύονται *ολόκληρα τα περιεχόμενα* των `session` & `flash`
  - Εναλλακτικά: αποθήκευση των συνεδριών του Rails σε πίνακα ΒΔ
  - (Αναζητήστε τη φράση "rails session use database table" στη Google)
  - Άλλη εναλλακτική: αποθήκευση των συνεδριών σε ένα σύστημα αποθήκευσης «NoSQL», όπως το *memcached*

69

70

70

Ο Ben Bitdiddle λέει: «Στο `session[]` μπορείτε να τοποθετήσετε οποιαδήποτε αντικείμενα (όχι μόνο «απλά» όπως οι ακέραιοι και οι συμβολοσειρές).» Εσείς τι πιστεύετε;

- ☐ Σωστό – εντελώς!
- ☐ Σωστό – αλλά κακή ιδέα!
- ☐ Λάθος, επειδή δεν μπορείτε να τοποθετείτε οποιαδήποτε αντικείμενα σε έναν κατακερματισμό
- ☐ Λάθος, επειδή το `session[]` δεν είναι ένας πραγματικός κατακερματισμός, απλώς «κραυγάζει» σαν τέτοιος

71

71

τέλος

72

72

Ολοκλήρωση με το CRUD

(Τεχνολογία ανάπτυξης λογισμικού ως υπηρεσίας §4.8)

73

Το ζεύγος Edit/Update είναι ανάλογο με το ζεύγος New/Create

- Τι είναι ίδιο;
  - Η 1η ενέργεια ανακτά τη φόρμα, η 2η ενέργεια την υποβάλλει
  - Το «submit» χρησιμοποιεί ανακατεύθυνση (για την ενέργεια `show` στην ταινία) αντί να απεικονίζει τη δική της προβολή
- Τι είναι διαφορετικό;
  - Η φόρμα θα πρέπει να εμφανίζεται με τις *υπάρχουσες* τιμές συμπληρωμένες:
    πρώτα ανάκτηση υπάρχουσας ταινίας http://pastebin.com/VV8ekFcn
  - Η ενέργεια φόρμας χρησιμοποιεί την `PUT` αντί της `POST`
    http://pastebin.com/0drjjxGa

Βοηθητική μέθοδος	Επιστρεφόμενο URI	Διαδρομή και ενέργεια RESTful	
movie_path(m)	/movies/1	PUT /movies/:id	update
movie_path(m)	/movies 1	DELETE movies/:id	destroy

74

Η destroy είναι εύκολη

- Θυμηθείτε, η destroy είναι μια μέθοδος *στιγμιοτύπου*
  - Πρώτα βρείτε την ταινία…μετά καταστρέψτε την
  - Στείλτε τον χρήστη πίσω στο Index

```
def destroy
  @movie = Movie.find(params[:id])
  @movie.destroy
  flash[:notice] =
       "Movie '#{@movie.title}' deleted."
  redirect_to movies_path
end
```

75

Αν ορίσετε μια μεταβλητή στιγμιοτύπου σε μια μέθοδο ελεγκτή, για πόσο θα διατηρηθεί η τιμή της;

- ☐ Για αυτή την αίτηση και όλες τις επακόλουθες αιτήσεις
- ☐ Μόνο για αυτή την αίτηση και την επόμενη
- ☐ Μόνο για αυτή την αίτηση – αφού απεικονιστεί η προβολή, η τιμή της μεταβλητής επανέρχεται σε nil
- ☐ Εξαρτάται από το αν η μεταβλητή στιγμιοτύπου δηλώθηκε ως στατική

76

76

τέλος

77

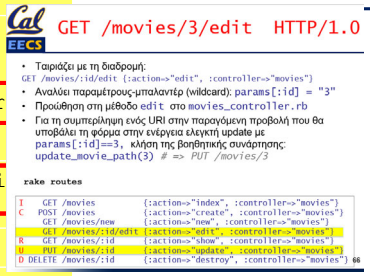
Περίληψη & σκέψεις: Αρχιτεκτονική SaaS, Rails, από το Sinatra στο Rails

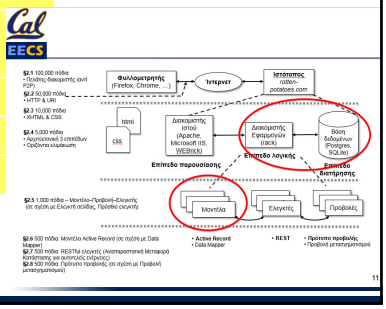
(*Τεχνολογία ανάπτυξης λογισμικού ως υπηρεσίας* §2.9-2.10, 4.11)

78

Σύνδεση αρχιτεκτονικών εννοιών με εφαρμογές του Rails

Gemfile	
Rakefile	
app	/models/, views/, controller
	/helpers
	/assets/stylesheets/applicati ss
config	/routes.rb
	/database.yml
db	/development.sqlite3 /test.sqlite3
	/migrate/
log	/development.log, test.log





79

Παγίδα: Διογκωμένοι ελεγκτές & κώδικας στις προβολές σας

- Πραγματικά εύκολο να πέσετε στην παγίδα των «διογκωμένων ελεγκτών»
  - Ο ελεγκτής είναι το πρώτο σημείο που επεξεργάζεστε στον κώδικα
  - Πειρασμός: να ξεκινήσετε τη συγγραφή κώδικα από τη μέθοδο ελεγκτή
- Διογκωμένες προβολές
  - «Το μόνο που χρειάζομαι είναι αυτό ο βρόχος for.»
  - «…και αυτός ο επιπλέον κώδικας που ταξινομεί τις λίστες ταινιών διαφορετικά.»
  - «…και αυτή η εντολή συνθήκης, σε περίπτωση που ο χρήστης δεν είναι συνδεδεμένος.»
- Όχι! Αυτά είναι για το μοντέλο, τον ελεγκτή, τις βοηθητικές μεθόδους

80

Μάθημα κώδικα μέρος 2

81

Μα πού, *επιτέλους,* **δεν** πρέπει να τοποθετήσετε κώδικα;

- Ελεγκτές
- Ελεγκτές
- Ελεγκτές
- Ελεγκτές

82

82

ΑΠΛΟΠΟΙΗΣΤΕ ΤΟΥΣ ΕΛΕΓΚΤΕΣ ΣΑΣ

ΑΛΛΙΩΣ ΠΑΕΙ ΤΟ ΚΟΥΝΕΛΙ

83

Η αρχιτεκτονική αφορά εναλλακτικές επιλογές

Υπόδειγμα που χρησιμοποιούμε	Εναλλακτικό
Πελάτης-Διακομιστής	Δίκτυο ομότιμων
Χωρίς κοινή χρήση (υπολογιστική νέφους)	Συμμετρικός πολυεπεξεργαστής, κοινόχρηστος καθολικός χώρος διευθύνσεων
Μοντέλο-προβολή-ελεγκτής	Ελεγκτής σελίδας, πρόσθιος ελεγκτής, πρότυπο προβολής
Active Record	Data Mapper
RESTful URI (όλη η κατάσταση που επηρεάζει την αίτηση είναι ρητή)	Το ίδιο URI κάνει διαφορετικά πράγματα ανάλογα με την εσωτερική κατάσταση

Καθώς θα δουλεύετε σε άλλες εφαρμογές SaaS εκτός αυτού του μαθήματος, θα εξετάζετε διαφορετικές αρχιτεκτονικές επιλογές και θα αναρωτιέστε για τις επιλογές.

84

84

Περίληψη: Από το Sinatra στο Rails

Δημιουργία εφαρμογής με `Gemfile`, `app.rb`, `config.ru`	Δημιουργία εφαρμογής με `rails new` *όνομα_εφαρμογής*· συγκεκριμένοι ρόλοι για διάφορους υποκαταλόγους της εφαρμογής
Δεν ορίζεται συγκεκριμένη αρχιτεκτονική εφαρμογής	Έντονη χρήση MVC, με χρήση ActiveRecord βασισμένο σε σχεσιακή ΒΔ για τα μοντέλα
Εμβόλιμες (inline) διαδρομές: `post '/new_game' do...end`	Ξεχωριστές περιγραφές διαδρομών στο `config/routes.rb`, υποστηρίζει εξ ορισμού διαδρομές πόρων CRUD RESTful, μπορούν να προστεθούν και άλλες
Αυθαίρετα ονόματα (εφαρμογή, μέθοδοι ελεγκτών, πρότυπα προβολών)	Σύμβαση αντί διευθέτησης
Χωρίς ενσωματωμένη συνδετικότητα βάσης δεδομένων	Πρόσθετο σύνδεσης (connector) σχεσιακής βάσης δεδομένων ActiveRecord και υποστήριξη για μεταβάσεις
Ίδια αρχεία, εξαρτήσεις, ρυθμίσεις, κ.λπ για ανάπτυξη & παραγωγή	Τρία περιβάλλοντα εξ ορισμού (ανάπτυξη, παραγωγή, έλεγχος)· μπορούν να οριστούν και άλλα
Τα πρότυπα προβολών εξαρτώνται από εσάς	Το MVC ορίζει μια προεπιλεγμένη ροή απεικόνισης, υποστηρίζει διάφορες μηχανές προτύπων
Εκκίνηση εφαρμογής με `rackup`	Εκκίνηση εφαρμογής με `rails server`

85

85

Από το Sinatra στο Rails (συνέχεια)

- Περισσότερα κινούμενα μέρη μεταξύ διαδρομής & απεικόνισης
  - Ταύτιση διαδρομής στο `config/routes.rb`
  - Κλήση κατάλληλης ενέργειας ελεγκτή
  - Εύρεση προτύπου προβολής με χρήση σύμβασης αντί διευθέτησης
  - Χρήση του `routes.rb` για τη δημιουργία διαδρομών στις προβολές
- Ωραία πράγματα που παρέχει το Rails σε σχέση με το Sinatra
  - `rails console` διαδραστικό περιβάλλον REPL
  - Διαδρομές που χρησιμοποιούν `PUT` & `DELETE`
  - Αυτόματη επαναφόρτωση κλάσεων όταν τροποποιείτε μια εφαρμογή
- Πράγματα που είναι ίδια
  - `session[]`, `flash[]`, `params[]`

86

86

Πλαίσια εργασίας, εφαρμογές, σχεδιαστικά υποδείγματα

- Πολλά σχεδιαστικά υποδείγματα μέχρι τώρα, θα εμφανιστούν και άλλα
- *Το 1995, ήταν άγρια δύση:* οι μεγαλύτεροι ιστότοποι ήταν μίνι υπολογιστές, *όχι* τρία επίπεδα/νέφος
- Βέλτιστες πρακτικές (υποδείγματα) «διαμορφώθηκαν» από την πείρα και ενσωματώθηκαν σε πλαίσια εργασίας
- Αλλά τα API το ξεπέρασαν αυτό: πρωτόκολλα του 1969 + γλώσσα σήμανσης της δεκαετίας του1960 + φυλλομετρητής του 1990 + διακομιστής Ιστού του 1992 δουλεύουν το 2011

87

87

Ποια βήματα απαιτούνται ΠΑΝΤΑ όταν προστίθεται μια νέα ενέργεια 'foo' στο μοντέλο Movie μιας εφαρμογής Rails:
(α) Διασφαλίστε ότι υπάρχει ένα πρότυπο για απεικόνιση στο `app/views/movies/foo.html.haml` (ή `.html.erb`, κ.λπ)
(β) Διασφαλίστε ότι υπάρχει μια διαδρομή στο `config/routes.rb`
(γ) Υλοποιήστε μια βοηθητική μέθοδο για τη δημιουργία των αναγκαίων βοηθητικών URI της διαδρομής

- ☐ Μόνο τα (α) και (β)
- ☐ Μόνο το (β)
- ☐ Μόνο τα (β) και (γ)
- ☐ Μόνο τα (α) και (γ)

88

89