

My Ruby on Rails Cheat Sheet

- Maria Stoica

1. Tutoriale

Incepe cu fast head start: (2-3 zile ca sa intelegi the basics f bine).

Rails for Zombies 1 & 2 <http://railsforzombies.org>

Apoi, un tutorial complet ca sa faci un site gen Twitter.

Rails Tutorial e-book <https://www.railstutorial.org/book>

2. Installing Rails

<http://rubyonrails.org/download/>

1. Install Ruby
2. Install Rails

3. Create a new project

```
$ rails new NumeProiect
```

```
$ bundle install # e indicat sa stergi Gemfile.lock si apoi sa dai bundle install
```

controller cu pagini unde Home e numele controllerului iar tot ce urmeaza dupa sunt numele paginilor (face cate un view pentru fiecare).

```
$ rails g controller Home page1 page2 page3
```

scaffold (model, controller, views)

User e numele modelului si al controllerului iar tot ce urmeaza dupa sunt attributele sub forma de nume_atribut:tip_atribut

tipuri de date: **string, integer, date, datetime**

```
rails g scaffold User name:string zi_nastere:date tag:integer
```

genereaza doar modelul (tabelul)

```
rails g model User name:string zi_nastere:date tag:integer
```

<http://guides.rubyonrails.org/migrations.html>

add column

```
rails g migration add_nume_atribut_to_table_name_plural nume_atribut:tip_atribut
```

Fiecare view are o metoda cu acelasi nume in controller-ul de care apartine. Cand creezi o pagina noua, ii adaugi si metoda in controller si o listezi si la routes. (ce e generat cu scaffold nui mai apare in routes cu toate view-urile ce tin de CRUD).

vreau sa lucrez cu `thing` in html. Il creez in controller dar e doar local dc il las `thing`.

Daca ii pun `@thing` atunci el e vizibil in view-ul corespunzator metodei.

```
def home
  cevaTempId = Altceva.where(ceva: false).first.pluck(:id)
  @thing = Thing.find(cevaTempId)
end
```

afisez numele lui `@thing` pe ecran. Ce vreau sa se vada pe ecran ii pun `<%=`

```
<%= @thing.nume %> <!-- asa arata codul ruby in view -->
```

daca nu ma intereseaza sa se afiseze il las fara =

```
<% if @things.count > 0 %>
<% end %>
```

4. Github Version Control

1. creezi un repo pe github si iti iei linkul

```
$ git init
```

```
$ git add .
```

```
$ git commit -m "message what this commit adds" # cu commit, schimbarile au loc doar local - daca le vrei si in repo-ul online at dai push
```

```
$ git remote add origin https://github.com/UserName/NumeProiect.git # remote-ul (repo-ul local), in cazul asta, se numeste origin
```

```
$ git push origin master # acum ce a fost retinut in commit este urcat online
```

```
$ git push origin master --force # dai --force atunci cand vrei sa inlocuiesti codul din repo-ul de pe github cu alt cod (in loc sa il modifici pe cel existent). De eg daca refaci aplicatia de la zero si nu vrei sa faci un repo nou
```

```
$ git remote -v # ca sa vezi ce remote-uri ai.
```

Github SSH trouble

<https://help.github.com/articles/generating-ssh-keys/>

5. The Data Base

“Skinny controllers, fat models.”

1. Crearea tabelelor si modificarea lor

- rails g model User name:string
- rails g scaffold User name:string
- rails g migration add_column_to_users column:string

2. Validarea atributelor in model (app/models/user.rb de eg)

http://guides.rubyonrails.org/active_record_validations.html

```
validates :content, length: {  
  minimum: 300,  
  maximum: 400,  
  tokenizer: lambda { |str| str.scan(/\w+/) },  
  too_short: "must have at least %{count} words",  
  too_long: "must have at most %{count} words"  
}
```

```
VALID_EMAIL_REGEX = /\A[\w+\-\.]+\@[a-z\d\-\.]+\.[a-z]+\z/i
```

```
validates :email, presence: true, format: { with: VALID_EMAIL_REGEX  
}, uniqueness: true
```

3. Querries in controller sau in view

http://guides.rubyonrails.org/active_record_querying.html

```
User.find(3)
```

unde 3 e id-ul

```
User.find_by_column_name("foobart")
```

unde "foobart" este valoare din column_name

```
User.where(tag:[1,5], isFree:true)
```

userii care sunt free si care au la coloana tag un numar cuprins intre 1 si 5

```
freeUserIds = User.where(isFree:true).pluck(:id)
```

din toti userii returnati de where pastrez doar id-urile in freeUserIds ca apoi sa pot face asta:

```
swordsOfFreePeople = Sword.where(user_id:freeUserIds)
```

acum am sabiile oamenilor liberi.

Daca, de eg, freeUserIds == [1, 2, 5, 34, 56] atunci

user_id:freeUserIds se traduce in SQL prin user_id IN [1, 2, 5, 34, 56]

```
@results = @results.where(este_libera: true).where("user_id IN  
(SELECT id FROM users WHERE specializare LIKE 'Ambele' OR  
specializare LIKE ?)", "#{get_current_user.specializare}")
```

poti sa pui si cod SQL in query. #{ruby code} inseamna ca ia valoarea dintre acolade si o pune in query

pentru editare - nu necesita sa dai .save dupa

```
User.find(3).update_attributes(name: "Bill", isFree: false)
```

stergi tot ce e in tabel

```
User.delete_all
```

stergi doar User.find(3)

```
User.find(3).destroy
```

6. Deploy to Heroku

<https://devcenter.heroku.com/articles/getting-started-with-rails4>

\$ heroku create

\$ heroku app:rename NewNameOfProject

\$ git push heroku master # push the app to heroku

\$ heroku run rake db:migrate # daca nu faci migrate dupa ce dai push prima oara da eroare

\$ heroku run rake db:seed # daca vrei sa populeze baza de date cu ce e in seed.rb

\$ heroku open # deschide aplicatia intr-o pagina noua

Each time you wish to deploy to Heroku:

\$ git add . -A

\$ git commit -m "commit for deploy to heroku"

\$ git push -f heroku

Consola bazei de date din heroku

\$ heroku run console

Bun de vazut erori desi e messy

\$ heroku logs

Heroku SSH trouble

<https://devcenter.heroku.com/articles/keys>

7. Useful Ruby Code

new or edit in a form

```
<%= form_for(@thing) do |f| %>
...
<% if @thing.new_record? == true %>
    <%= f.hidden_field :isFree, :value => false %>
    <% end %>
...
<%end%>
```

```
# form helpers
```

http://guides.rubyonrails.org/form_helpers.html

```
text_field
number_field
date_field
email_field
hidden_field
```

http://guides.rubyonrails.org/form_helpers.html

```
text_field
number_field
date_field
email_field
hidden_field
```

```
# parcurgerea unui set
```

```
@things.each do |thing|
  ...
end
```

```
# timp ramas in cuvinte (in engleza) - primeste datetime sau date
```

```
time_ago_in_words(get_current_sesiune.data_start)
```

corectarea pluralului - doar sa puna "s" la final pt plural

```
config/initializers/inflections.rb
ActiveSupport::Inflector.inflections do |inflect|
  inflect.irregular 'a', 'as'
end
```

```
# vezi continutul variabilelor cu p si puts
```

Cand dai **\$ rails s** iti apar in terminal toate activitatile site-ului si ce requesturi au fost facute si ce s-a raspuns la ele. (WEBrick)

Pe heroku ai **\$ heroku logs**

Ca sa afisezi ceva in terminal poti folosi p sau puts in controller sau in view.

```
puts "smth" + self.first_name + thing.numar.to_s
```

sau

```
p foo
```

care e acelasi lucru cu

```
puts foo.inspect
```

<http://stackoverflow.com/questions/1255324/p-vs-puts-in-ruby>

mail setup (partial - pasii mari)

http://guides.rubyonrails.org/action_mailer_basics.html

Setting the email in the app

config/environments/development.rb

```
config.action_mailer.delivery_method = :smtp
  config.action_mailer.smtp_settings = {
    address:           'smtp.gmail.com',
    port:              587,
    domain:             'gmail.com',
    user_name:         'fmiunivbuc@gmail.com',
    password:          'fmiunibuc',
    authentication:    'login',
    enable_starttls_auto: true  }
```

config/routes.rb

```
match "intreaba/send", to: "browse_pagini#sendmail", via: "post"
```

app/mailers/user_mailer.rb

```
class UserMailer < ActionMailer::Base
  default from: "fmiunivbuc@gmail.com"

  def notification_email(from_mail, to_mail, subject, text)
    @text = text
    mail(from: from_mail, to: to_mail, subject: subject)
  end
end
```

Send the email

method in a controller

```
def sendmail

  UserMailer.notification_email(User.find(get_current_user.id).e
    mail, params[:to], params[:title], params[:text]).deliver
end
```



```
    redirect_to root_path  
  
end
```

setting up your own authentication (partial - pasii mari)

(de la capitolul 6 la capitolul 8)

https://www.railstutorial.org/book/modeling_users#sec-adding_a_secure_password

Gemfile

```
gem 'bcrypt-ruby', '3.1.2'
```

\$ bundle install

adauga atributul password_digest in User

\$ rails g migration add_password_digest_to_users password_digest:string

app/models/user.rb

```
has_secure_password
```

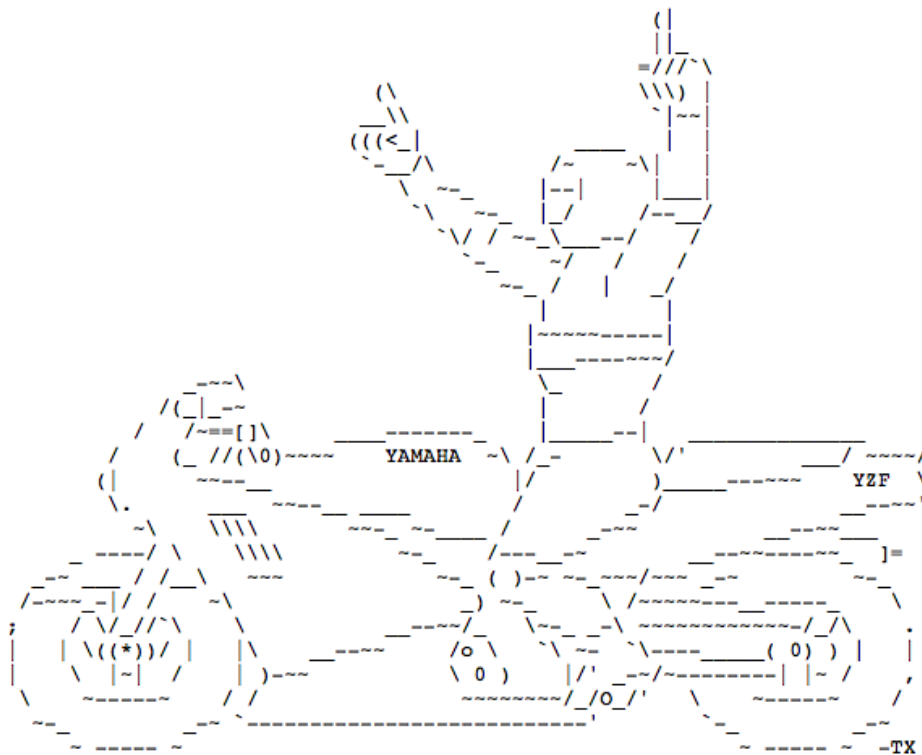
\$ rails c

> User.create(name:"Tom", password:"foobar", password_confirmation:"foobar")

8. Next Steps

- CodeSchool's Ruby Path: <https://www.codeschool.com/paths/ruby>
 - Rails 4: Zombie Outlaws: <http://rails4.codeschool.com/videos>
 - Rails 4 Patterns <https://www.codeschool.com/courses/rails-4-patterns>
 - Surviving APIs with Rails
<https://www.codeschool.com/courses/surviving-apis-with-rails>
 - Rails Testing for Zombies:
<https://www.codeschool.com/courses/rails-testing-for-zombies>
 - Testing with RSpec: <https://www.codeschool.com/courses/testing-with-rspec>
 - Ruby Bits: <https://www.codeschool.com/courses/ruby-bits>
 - Ruby Bits 2: <https://www.codeschool.com/courses/ruby-bits-part-2>
- Places with Books:
 - This is sheer awesomeness: <http://it-ebooks.info>

- Have fun!



- Imagination is the limit.