

Ruby on Rails

Lexi Ross | lbross@college.harvard.edu

IT'S CATURDAY?



What is Ruby on Rails?

- ▶ Ruby on Rails, or RoR, is a **web application framework** that runs on the **Ruby programming language**
 - ▶ **What is a framework, and why might it be useful to use one?**
- ▶ **Ruby** is a dynamically typed, objected oriented programming language
- ▶ **Rails** is organized around the Model-View-Controller architecture
- ▶ **Agenda: Ruby, then Rails**





Ruby

Language of champions

Why Ruby?

- ▶ Powerful and easy to use (example: sort a list)
 - ▶ C: [an implementation of mergesort, taking up many lines]
 - ▶ Ruby: `array.sort`
- ▶ Code is easy to read and parse because it looks more like English than C does
 - ▶ Comments aren't as necessary
- ▶ Lots of libraries (“gems”) to extend functionality
- ▶ Plenty of help online
- ▶ It's FUN!



Getting Started

- ▶ Start up your Appliance and open a Terminal window
- ▶ Type `irb` to start an interactive Ruby session
- ▶ You should see a command line prompt!



Ruby Syntax

- ▶ No type declarations, can mix types (like in PHP)
- ▶ `x = 5`
- ▶ C:
 - ▶

```
if(x < 10)
    printf("The number is %d", x);
```
- ▶ Ruby (both of these work):
 - ▶

```
puts "The number is #{x}" if x < 10
```
 - ▶

```
if x < 10
    puts "The number is #{x}"
end
```
- ▶ `puts == put string` (like `echo` in PHP)
- ▶ Can put conditional statements **AFTER** code block if desired



Arrays and hashes

- ▶ `nums = [1, 2, 3, 4, 5]`
- ▶ `nums << "hello"`
- ▶ Arrays can be of mixed types and can be dynamically resized
- ▶ There are numerous built-in methods for arrays
 - ▶ `sort`, `shift`, `reverse`, `shuffle`, etc.
- ▶ A hash is an associative array (like a map in PHP)
 - ▶ `cat = { 'name' => 'Pepper', 'age' => 6 }`
 - ▶ `cat['color'] = 'black'`



Loops

- ▶

```
words = ['The', 'cat', 'jumped', 'over', 'the',  
        'moon', 'on', 'Monday']  
words.each do |w|  
    puts w if w.upcase.start_with?('MON')  
end
```
- ▶ This code prints out all words beginning with the letters “mon” (case insensitive)



Ruby + HTML

- ▶ Can integrate in the same way we insert PHP code into HTML
- ▶ Files carry the extension `.html.erb`
 - ▶ Just `ruby:.rb`
- ▶

```
<% if x > 0 %>  
    <div>The number is <%= x %>.</div>  
<% end %>
```



More Ruby resources

- ▶ <http://www.ruby-doc.org/>
- ▶ <http://www.ruby-lang.org/en/documentation/>
- ▶ <http://stackoverflow.com/>





Rails

Not your grandmother's web application framework

Why Rails?

- ▶ The MVC framework makes it easy to separate the different functional layers of your application
- ▶ Very popular right now – it's great to know the latest technologies that are shaking things up!
- ▶ Easy to get started creating a new application right away



Model-View-Controller Framework

- ▶ A way of organizing components of a web application
- ▶ Separates the internal application logic from the user interface
- ▶ **Model:** the “application logic”
 - ▶ Generally, each table in your database has a corresponding model in your application
 - ▶ Methods for extracting information from your database
- ▶ **View:** the “front end”
 - ▶ What the user actually sees
 - ▶ Includes .html.erb files
- ▶ **Controller:** the “mastermind”
 - ▶ Interacts with both the user and the models
 - ▶ Process incoming user input; access models for data; returns to the user with the data requested
 - ▶ Defines instance variables that will be accessed in the view



Let's get started with a Rails app!

- ▶ In Terminal:
 - ▶ `sudo gem install rails`
 - ▶ `rails new [application name]`
- ▶ This creates a “skeleton” app
- ▶ Problems with installing Rails in the Appliance? You can also use your own computer!
 - ▶ Mac: Terminal
 - ▶ PC: PuTTY
- ▶ We also need to create a database!
 - ▶ Set up



Generating new components

- ▶ `rails generate scaffold User username:string password:string cash:decimal`
- ▶ `rake db:migrate`

- ▶ Scaffolding generates a model with the given attributes
- ▶ For step-by-step instructions creating a Rails app from scratch, check out http://guides.rubyonrails.org/getting_started.html



Models vs. MySQL

- ▶ No need for SQL queries in Rails!
- ▶ MySQL **rows** become Ruby **objects**, and MySQL **columns** become **attributes of those objects**
- ▶ Say we have a table called `users` with a column called `cash`, and we have a variable called `current_user`
 - ▶ `current_user.cash`
 - ▶ Single line of code accomplishes what would have been several lines of PHP!



Testing your app

- ▶ `rails console`
 - ▶ Allows you test snippets of code (similar to irb)
 - ▶ Gives you access to your database
- ▶ `rails server`
 - ▶ Point your browser to `localhost:3000/`



Jumping ahead...

- ▶ Let's check out a more fleshed out Rails app
- ▶ Source: http://pragprog.com/titles/rails4/source_code

- ▶ Things to note
 - ▶ :symbol vs. 'string'
 - ▶ Database migrations (up vs. down)
 - ▶ has_many vs. belongs_to



Why not Ruby on Rails?

- ▶ Ruby tends to be a slower language, so there are issues with scalability
 - ▶ cf. Twitter
- ▶ Rails is not the ideal framework when working with a large number of complex models, as switching between model files becomes tedious
- ▶ Configuration can be a bit tricky at times, especially when you're dealing with different versions
- ▶ But for a CS50-scale project, it's pretty awesome!



Recommended Reading

- ▶ *Agile Web Development with Rails (4th edition)*
 - ▶ <http://pragprog.com/book/rails4/agile-web-development-with-rails>
- ▶ Source code of Rails projects!
- ▶ <http://guides.rubyonrails.org/index.html>



That's all, folks!



Questions?

