Web Development and Ruby on Rails

A quick look at a modern full-stack web application development framework

Who am I?

- Mark Brooks
- Senior Web Developer at Arthrex, Inc.
- Working on Surgical Outcome System web application.
- Working with Ruby on Rails since 2006.

What is Ruby?

- Ruby is a dynamic, interpreted, reflective, object-oriented, general-purpose programming language.
- Developed by Yukihiro “Matz” Matsumoto.
- Publicly released in 1995.
- https://www.ruby-lang.org/en/
- TryRuby: https://ruby.github.io/TryRuby/

- Dynamic: Types are verified when the program is run. Ruby employs duck-typing; the type of an object is determined by methods and properties, i.e. what it does and what data it stores.

- Interpreted: Ruby programs are run as written by an interpreter, rather than being compiled beforehand into a code closer to what the machine understands.

- Reflective: Ruby programs can examine and modify themselves or other programs at runtime.

- Object-Oriented: Ruby programs support modeling problems as classes consisting of data and actions on that data, instances of which, called objects, are used in the program.

“Ruby is simple in appearance, but is very complex inside, just like our human body.”

–Yukihiro “Matz” Matsumoto

What is Ruby on Rails?

- Ruby on Rails (ROR) is a web application framework written in Ruby.
- Ruby on Rails utilizes model-view-controller (MVC), convention over configuration (CoC), don’t-repeat-yourself (DRY) and the ActiveRecord pattern for database access.
- Developed by David Heinemeier Hansson
- Publicly released in July 2004, as an extraction of the framework code underlying the Basecamp application.
- http://rubyonrails.org

- Web application framework: A framework for writing client-server programs where the client is a web browser. HTML, CSS, Javascript, + data and actions.

- Model-View-Controller: A user interface architecture used to separate data and operations (model) from their representation (view) via an interpreter layer (controller).

- Convention over Configuration: Most decisions are made for the programmer using sensible defaults, so only the unconventional must be defined.

- Don’t Repeat Yourself: Information is located in a single unambiguous place. Reduce repetition with methods, subroutines, code generators and introspection.

- ActiveRecord Pattern: Object-relational database mapping.

“The best frameworks are in my opinion extracted, not envisioned. And the best way to extract is first to actually do.”

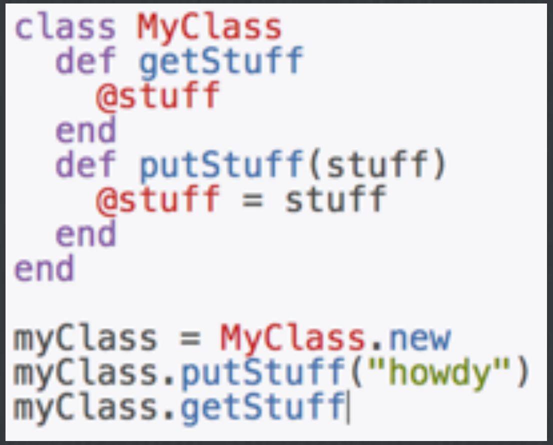
–David Heinemeier Hansson

Your first Ruby program

- ☐ Go to the following address in your web browser: https://ruby.github.io/TryRuby
- ☐ Click the “Clear” button.
- ☐ In the Editor field, type: `puts “Hello from Ruby!”`
- ☐ Click the blue “Run” button.

Your first Ruby class

- After clearing the Editor, type the following, then hit "Run":

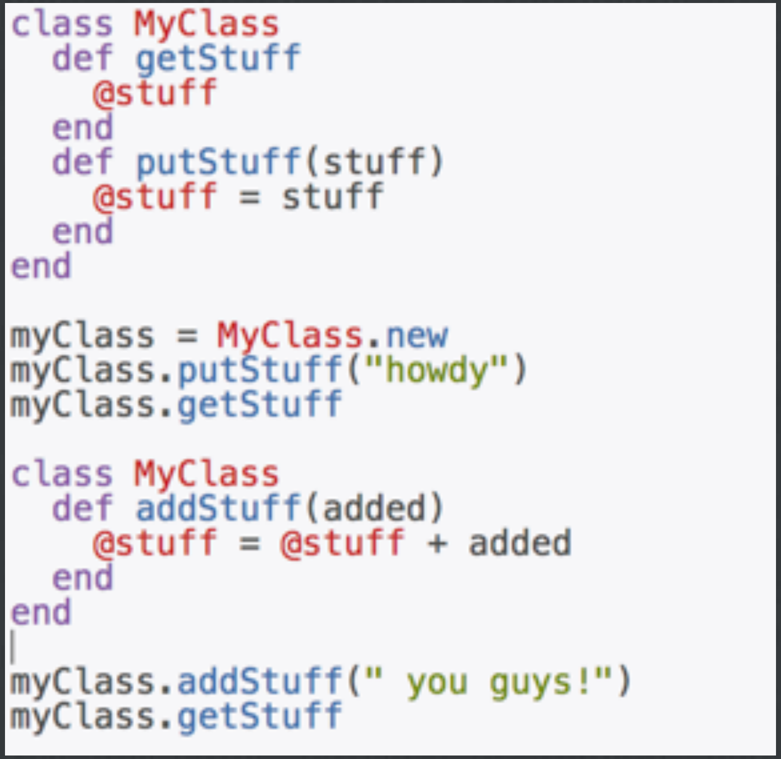


```
class MyClass
  def getStuff
    @stuff
  end
  def putStuff(stuff)
    @stuff = stuff
  end
end

myClass = MyClass.new
myClass.putStuff("howdy")
myClass.getStuff
```

- Notice that there are no type declarations. It just works.

Ruby classes are open

- Change the same code you used in the editor last time to look like this:
- (and click "Run")

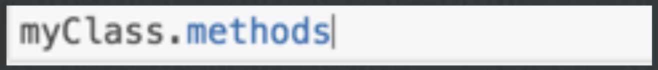


```ruby
class MyClass
  def getStuff
    @stuff
  end
  def putStuff(stuff)
    @stuff = stuff
  end
end

myClass = MyClass.new
myClass.putStuff("howdy")
myClass.getStuff

class MyClass
  def addStuff(added)
    @stuff = @stuff + added
  end
end

myClass.addStuff(" you guys!")
myClass.getStuff
```

- Being a dynamic interpreted language, Ruby classes can be modified after creation.

Ruby is introspective

- Add the following line at the end of the program you have in your editor and click “Run”:



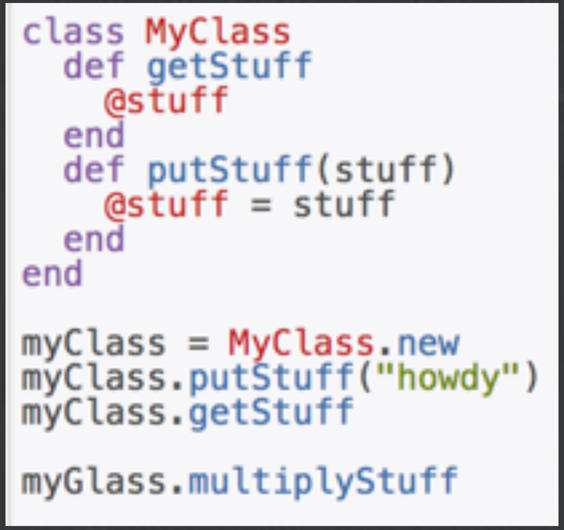
```
myClass.methods
```

- You get back a list of methods, many of which you didn’t actually create. That’s because Ruby is designed to be introspective at runtime.

- Your class implicitly inherits from a parent class defined in the language runtime which implements methods, like “methods”, that make it possible to identify the state of the running program in the machine.

Ruby throws errors

- Now let's "clear" your editor and make your code look like this:

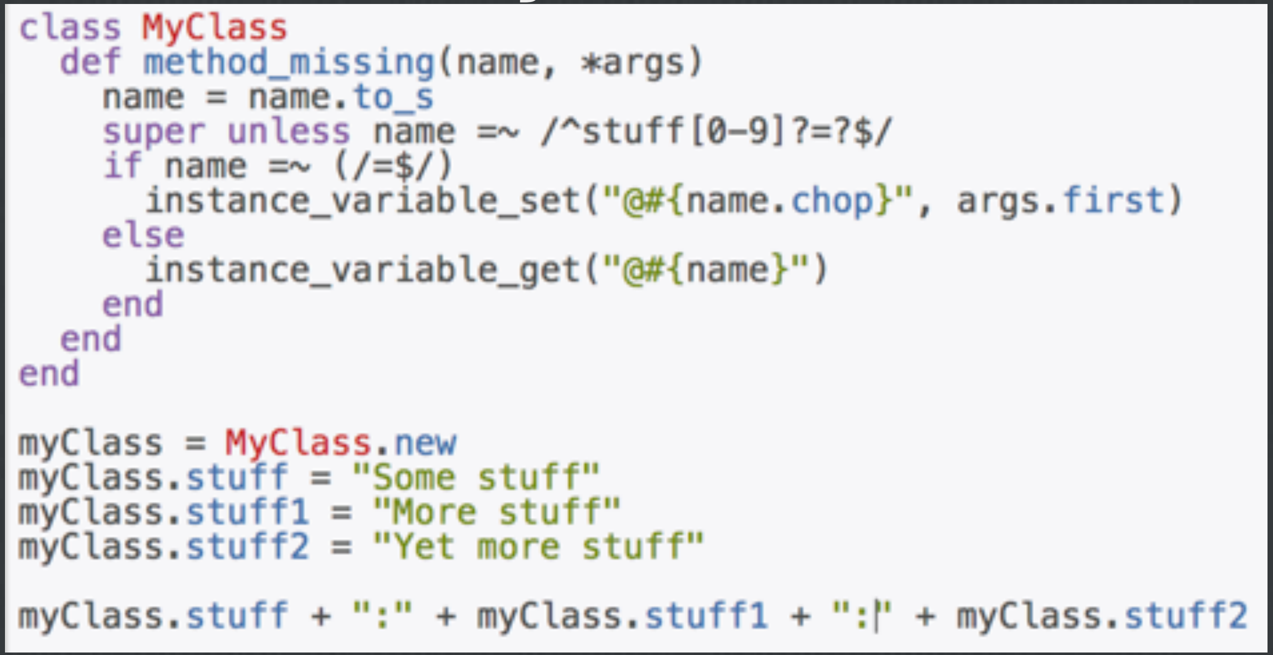


```
class MyClass
  def getStuff
    @stuff
  end
  def putStuff(stuff)
    @stuff = stuff
  end
end

myClass = MyClass.new
myClass.putStuff("howdy")
myClass.getStuff

myGlass.multiplyStuff
```

- Click "Run". What happens?
- So what happens when you combine this with introspection?

Ruby Metaprogramming

- Clear your Editor and make your code look like this, then click “Run”:



```ruby
class MyClass
  def method_missing(name, *args)
    name = name.to_s
    super unless name =~ /^stuff[0-9]?=?$/
    if name =~ (/=$/)
      instance_variable_set("@#{name.chop}", args.first)
    else
      instance_variable_get("@#{name}")
    end
  end
end

myClass = MyClass.new
myClass.stuff = "Some stuff"
myClass.stuff1 = "More stuff"
myClass.stuff2 = "Yet more stuff"

myClass.stuff + ":" + myClass.stuff1 + ":" + myClass.stuff2
```

- Your output should look like:

```
Some stuff:More stuff:Yet more stuff
```

- You’ve effectively added code dynamically to your program. This is very DRY.

Why does meta-programming matter in Ruby?

- Being able to program your programming language means you can save lots of developer time that would otherwise be spent writing what is known as “boilerplate code”.
- Ruby On Rails is built around this feature of Ruby

Ruby on Rails examples

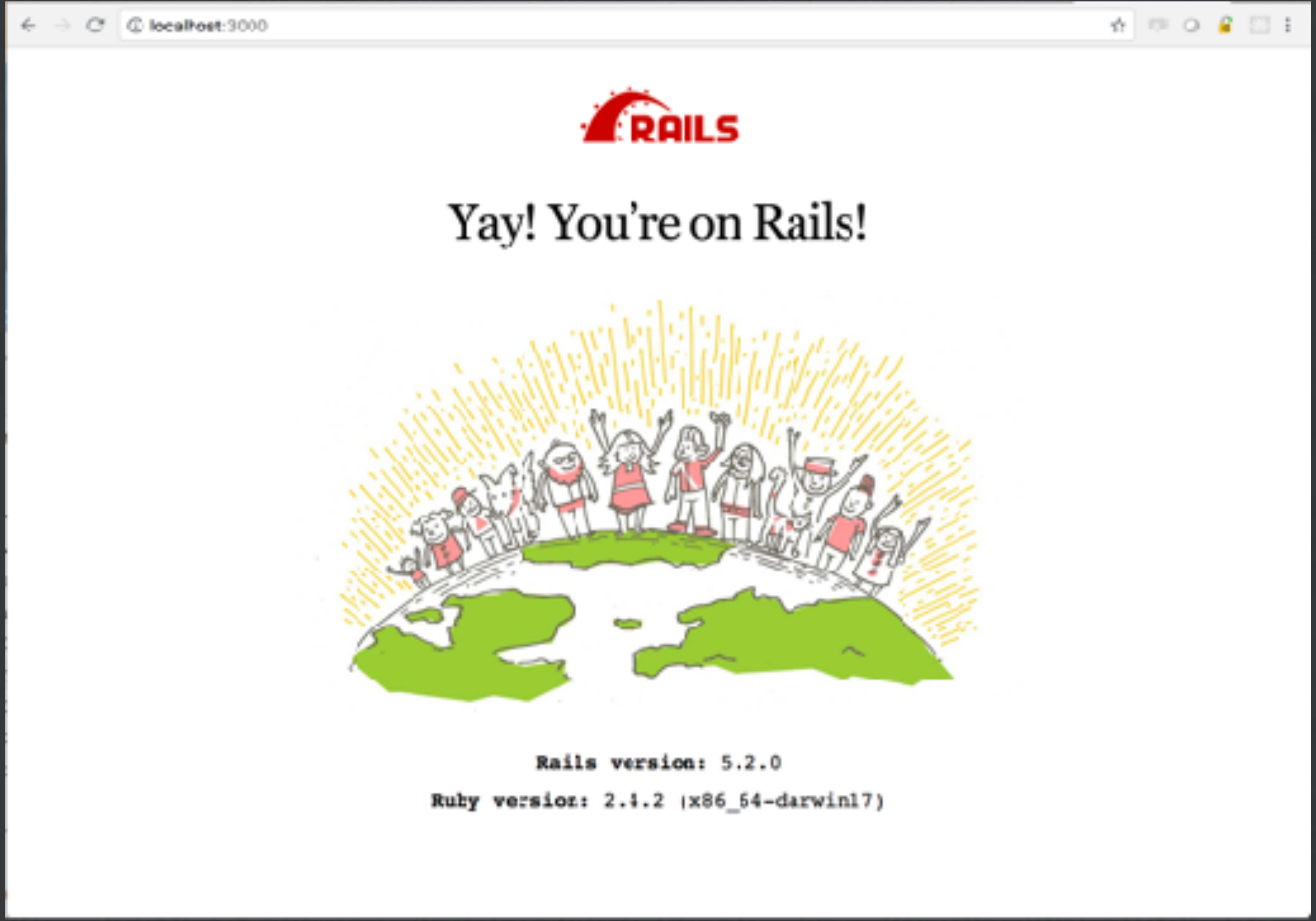
- installrails.com is one way to easily install Ruby on Rails on your computer.
- If you have RubyGems installed, “gem install rails” should get you the latest libraries constituting Rails.
- This demo won’t require you to install anything for now.

Generate a Rails application

- To create an "Ice Cream Store" application, we might do the following: `rails new icecreamstore`
- You will get a LOT of output and a new directory called "icecreamstore".
- Change into that directory, and run the following command: `rails server`

Generate a Rails application 2

- Point your web-browser to localhost:3000 (or 127.0.0.1:3000) and tada!



Yay! You're on Rails!

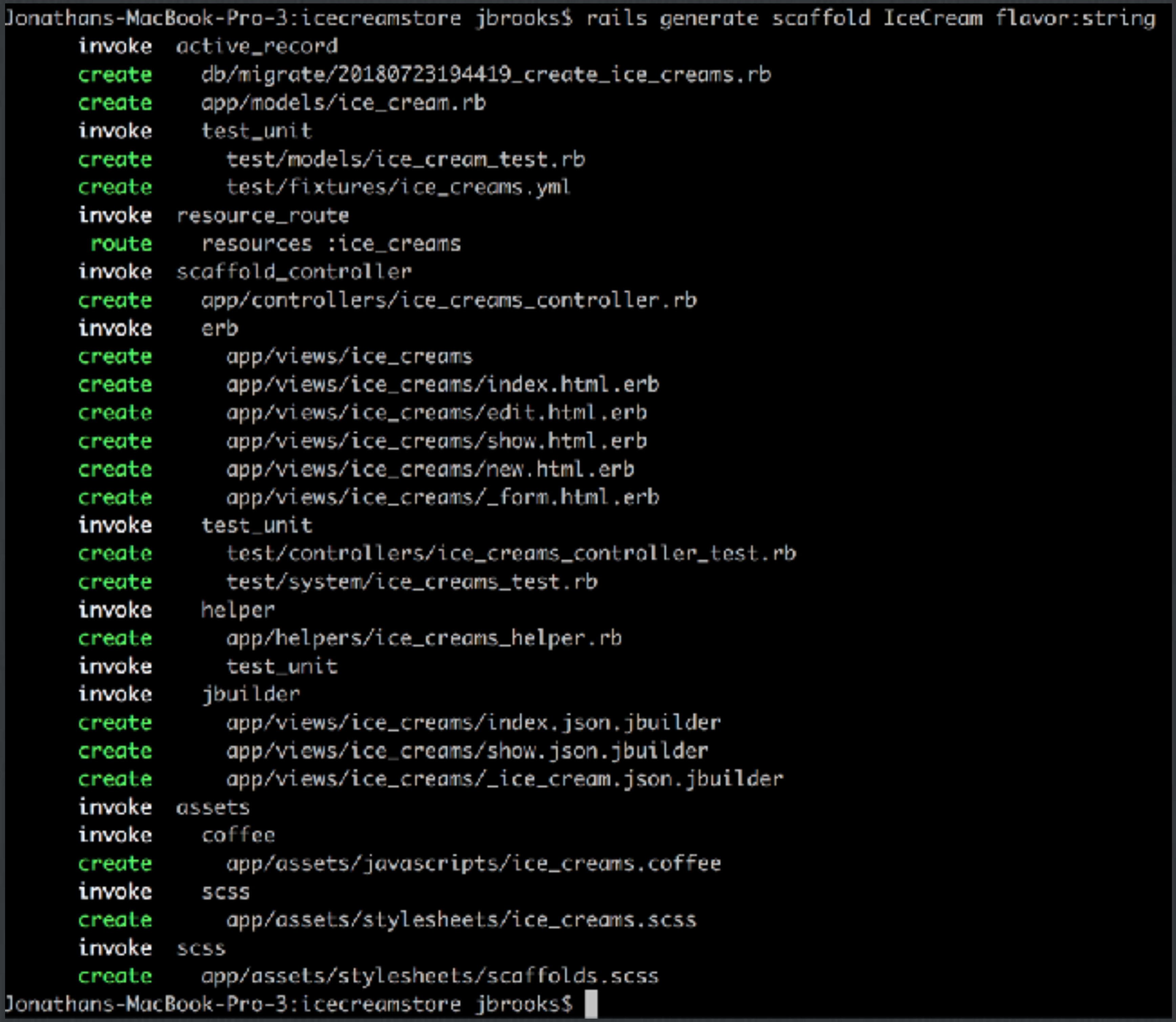
```
Rails version: 5.2.0
Ruby version: 2.4.2 (x86_64-darwin17)
```

-

Saving time with generators

- ☐ We know that we need to be able to sell ice cream, and that the ice cream will come in various flavors, prices, etc. Let's model that and let Rails make it easier to do for us by using the Rails generators.
- ☐ To give us a starting point, let's create something called a <u>scaffold</u>. `rails generate scaffold IceCream flavor:string`

- Output!



```
Jonathans-MacBook-Pro-3:icecreamstore jbrooks$ rails generate scaffold IceCream flavor:string
      invoke  active_record
      create    db/migrate/20180723194419_create_ice_creams.rb
      create    app/models/ice_cream.rb
      invoke    test_unit
      create      test/models/ice_cream_test.rb
      create      test/fixtures/ice_creams.yml
      invoke  resource_route
       route    resources :ice_creams
      invoke  scaffold_controller
      create    app/controllers/ice_creams_controller.rb
      invoke    erb
      create      app/views/ice_creams
      create      app/views/ice_creams/index.html.erb
      create      app/views/ice_creams/edit.html.erb
      create      app/views/ice_creams/show.html.erb
      create      app/views/ice_creams/new.html.erb
      create      app/views/ice_creams/_form.html.erb
      invoke    test_unit
      create      test/controllers/ice_creams_controller_test.rb
      create      test/system/ice_creams_test.rb
      invoke    helper
      create      app/helpers/ice_creams_helper.rb
      invoke      test_unit
      invoke    jbuilder
      create      app/views/ice_creams/index.json.jbuilder
      create      app/views/ice_creams/show.json.jbuilder
      create      app/views/ice_creams/_ice_cream.json.jbuilder
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/ice_creams.coffee
      invoke    scss
      create      app/assets/stylesheets/ice_creams.scss
      invoke  scss
      create    app/assets/stylesheets/scaffolds.scss
Jonathans-MacBook-Pro-3:icecreamstore jbrooks$ 
```

Rails lays the foundation

- Notice that by running one line, we generated a lot of files. Ruby files. CSS (stylesheet) files. Javascript files. HTML. Routing. All this only took one command line.

- The purpose of the generators is to give you a quick and easy way to get the things you need to do to get the program up and working done quickly.

- Notice how there is a specific structure to where files of a certain kind go. That's part of the Rails CoC approach — you don't have to write an XML file to tell the server where to look for things so long as they are located in the standard places.