- Use **rails new** to generate a new web application. Then **cd** into the new subfolder.
- Once you **cd** into the app's folder, all subsequent commands should be run from that folder
- If you ever clone an existing Rails app from GitHub, run **bundle install** inside the app's folder
- Use **rails server** to start a local HTTP server on http://localhost:3000
- Press Ctrl-C to stop the server. (Try to refresh your browser and see what happens)
- Learn the ***RCAV™*** recipe for creating web pages in Rails: **R**oute-**C**ontroller-**A**ction-**V**iew
- Rails expects you to know the MVC design pattern
- Rails expects you to know HTTP philosophy
- Every URL must be defined in routes.rb
- Every URL mapped to a controller class and action method
- Controller classes are Ruby classes derived from ActionController::Base
- Actions can respond with direct HTML or use an .html.erb view file
- Views can be plain HTML or HTML with embedded Ruby (.erb)
- HTML elements consist of tags, attributes, and sometimes content
- Elements that require content must also contain a closing tag
- Action methods can create @vars for the view to use
- HTTP is stateless!
- URLs can contain querystring parameters
- Rails lets you access querystring parameters via the params hash
- **<form>** and **<a>** let the user navigate to other pages
- The Gemfile lists 3rd-party tools and dependencies
- If you ever need to change your Gemfile, run **bundle install**

```
get '/flicks', controller: 'movies', action: ‘list'
```

Consider The Following…

The data in the Movies app is hardcoded, in the sense that we created the data, not the users of our website.

Do the websites you use generally rely on hardcoded data?

What happens to that data when you close your browser?

Suppose we are building Amazon.com, and we identify that we will need have a *products* resource. We will need a controller to manage that resource.

Here is a simple 3-Step Recipe:

1. Create a new file: **`app/controllers/products_controller.rb`**

   Inside the file, define a Ruby class that derives from **`ApplicationController`**:

   ```
   class ProductsController < ApplicationController

   end
   ```

2. Create a new views subfolder: **`app/views/products/`**

3. There is no step 3. :-)

Controllers are Always Pluralized!

ProductsController

not

ProductController

config/routes.rb

```
get 'movies' => 'movies#index'

# or if you prefer:
# get 'movies', :controller => 'movies', :action => 'index'
```

app/controllers/movies_controller.rb

```
class MoviesController < ActionController::Base

  def index
     @titles = ["Apollo 13", "Star Wars"]
  end

end
```

app/views/movies/index.html.erb

```
<h1>Movies</h1>

<ul>

<% @titles.each do |movie_title| %>

   <li><%= movie_title %></li>

<% end %>
```