Introduction to Rails

Our Goals

- Understand the history of Rails
- Get Rails installed
- Get a feel for the principles of Rails
- Be able to talk about the MVC structure
- Build our first application

History of Ruby

- David Heinemeier Hansson is the creator
  - Twitter
  - Github
  - Medium
- Started as a part of BaseCamp
- Released as Open Source in July, 2004
- Commit rights given out in February, 2005

Important Links

- The Rails Doctrine
- The Ruby on Rails website
- The Ruby on Rails guides
- A study of the Rails application structure

What is it?

- It is a web application framework
  - Gives structure
  - Saves repetition
- It is quite opinionated
  - It makes assumptions, there are lots of ways to do things and it is flexible - but it will make it a lot easier if you follow convention
- The Ruby on Rails Doctrine

The Guiding Principles

Don't Repeat Yourself (D.R.Y)

- "Every piece of knowledge must have a single, unambiguous, authoritative representation within a system"
- Write once, use it everywhere
- Makes code more maintainable, extensible, less buggy and less surprising

The Guiding Principles

Convention over Configuration (C.o.C)

- Rails is quite opinionated
- If you don't follow the guidelines that it sets out (naming conventions etc.), it will be a lot more difficult
- Some things may feel like absolute magic

The Guiding Principles

Models, View and Controllers (M.V.C)

- This is probably the most important
- MVC is a software architecture pattern. It is an attempt to:
  - Break code down into manageable chunks
  - Make it easy for developers to work together
- It has been around for a long time

The Guiding Principles

Models

- The model manages the behavior and data of the application domain
- It is also where the database classes are created
- It is often a direct reflection of a table
- It is meant to be where all of the "business logic" exists
- The model directly manages the data, logic and rules of the application

The Guiding Principles

Views

- Manages the display of information
- Gets given information by the controllers
- A *view* can be any output representation of information

The Guiding Principles

Controllers

- The controller interprets the user behaviour
- Acts as the glue between the views and the models
- Makes changes to the models, palms details off to the views

The Guiding Principles

MVC

- The model is the data
- The view is the window on the screen
- The controller is the glue between the two

The Guiding Principles

It's all about interactions

- A controller can send commands to the model to update the model's state
- It can also send commands to the associated view to change the presentation (and content)
- A model stores data that is retrieved by the controller and displayed in the view
- Whenever there is a change to the data it is updated by the controller
- A view requests information from the model that it uses to generate an output representation to the user

Let's install it

```
gem install rails
```

Let's create a Rails app

```
rails new some_app_name

cd some_app_name
```

The Rails Directory Structure

These are the ones we will be caring about today

- app
  - assets
  - controllers
  - models
  - views
- config
  - routes
- Gemfile

The approach

1. rails new app_name
2. Edit Gemfile for debugging and run bundle
3. Work out config/routes.rb
4. Create controllers
5. Add methods to controllers
6. Create appropriate views
7. Repeat as necessary

Your homework