Ruby on Rails Bootcamp

Ruby on Rails allows you to quickly build dynamic web applications. Mastering Ruby on Rails has two parts: the Ruby Programming language and the Rails framework. Learn both Ruby and Ruby on Rails and build full-functioning web applications from your first class.

Group classes in NYC and onsite training is available for this course. For more information, email corporate@nobledesktop.com or visit: https://www.careercenters.com/courses/ruby-bootcamp-nyc

nyc@careercenters.com • (212) 684-5151

Course Outline

Introduction to Ruby on Rails

Scaffolding
- The Scaffold command
- Overview of Models, Views, and Controllers (MVC)

Adjusting the Templates Created by Scaffolding
- Formatting in Rails
- Working with Dynamic web pages
- Editing the text in the tab
- Redirecting the homepage URL
- Editing the CSS

Version Control with Git
- Initializing a repository
- Commiting and Pushing changes
- Creating, switching, and deleting branches
- Merging branches

Ruby Fundamentals

Ruby Data Types & Variables
- String, Integer, Float, Boolean and Nil values
- Properties of Ruby data types
- Instance variables & Local variables
- Global variables

Functions & Control Flow

- Built-in functions

- Creating your own functions

- Passing arguments and returning values

- If/Else and Unless Statements

- While/Until Loops

Ruby Data Structures

- Arrays: The Simplest Collections

- Hashes

- Enumerators

- Common Iterators

Classes

- Creating classes

- Inheritance

- Class Methods

- Overriding Methods

Controllers and Views

Generating a Controller

- Creating a New Rails Site for Flix

- How Controller methods relate to views

- Private methods

- The params hash

Views

- Generating and creating Views

- When you don't need a View

- Mapping Views to controller actions and routes

- Dynamic Views

- The rails routes command

Models & Forms

Models

- Generating a Model

- How Migration Files work

- Migratiing the Database

- Rolling back a migration

Rails Forms

- Rails forms vs. HTML forms

- HTTP Overview

- Rails Form Helpers

- Rails forms: form_for, form_tag, and form_with

- Connecting a form to a Model

Uploading Files

- Installing ActiveStorage
- Allowing users to upload images

Views

- Creating a View
- Adding Dynamic Data
- Rendering a Partial
- Optional Bonus: Rendering a View

Advanced Models

Model Validations

- The purpose of validations
- Adding basic validations
- Preventing submission of empty forms
- Customizing validations
- Adding Error Messages

Model Methods

- Built-in Model methods
- Adding methods to models

Model Relationships

- Has_one and belongs_to relationships
- Has_and_belongs_to_many: Simple Many-to-Many Relationships
- Has_many, through: Advanced Many-to-Many Relationships with Additional Metadata
- Polymorphic Relationships

Other Important Relationships in Rails

- Delegates: Sharing Methods Between Related Objects
- Self-Joins: Relationships Between Instances of the Same Model

ActiveRecord Associations

Model Relationships

- Has_one and belongs_to relationships
- Has_many relationships
- Joining models together via the controller
- Writing Simple Tests Using Fixtures
- Optional Bonus: Writing Tests Using Embedded Ruby Code
- Additional Bonus: Helpers

Integrating Front-end Code

Rails Asset Pipeline

- What the Asset Pipeline does

- Adding CSS
- Adding JavaScript
- Conditional CSS and JavaScript files

Launching an Application

Preparation
- Managing Ruby environments
- Locking Gem versions
- Environment variables

Amazon Web Services
- Setting up an AWS account
- Where to store credentials
- Remote Asset Storage with Amazon S3

Heroku
- Heroku Account Setup
- Installing Heroku Toolbelt
- How Heroku works
- Types of Dynos
- Deploying your application