// FLATIRON SCHOOL

WEB DEVELOPMENT IMMERSIVE SYLLABUS

The goal of this course is to enable students to pursue a career in Web Development by providing them with the baseline skills and experience necessary to obtain entry-level jobs. The curriculum is designed to teach students to think like software engineers, independent of any specific language. That said, the bulk of the material covered is based on the Ruby ecosystem.

Students develop a foundation in programming fundamentals, and conquer the concepts of object-oriented programming. Students will work with APIs (Application Programming Interfaces), become proficient in database modeling and ORM (Object Relational Mapping), understand the concept of MVC the (Model View Controller) Framework, and execute application deployment. Labs are taught using test-driven development, allowing students to gain a real-world programming experience, while giving instructors the ability to evaluate and assist students in real time.

By the end of the semester, all students build functional web-based applications. Students also expand their digital portfolios by maintaining active technical blogs and giving technical presentations to their classmates and the public via "Meetups."

Why Ruby?

We will leverage the Ruby programming language to deliver these concepts for three primary reasons:

1- **Readability**- Much of the initial difficulty in learning programming stems from the learning curve necessary to gain comfort with a language's syntax. While traditional languages like C++, Java, and even python, employ white-space sensitivity and heavy use of constructs like brackets and semi-colons, Ruby is designed to be more readable and accessible, allowing new programmers to focus immediately on the fundamental concepts and logic, rather than basic syntax.

2- **Open Source**- The Ruby language has nurtured an incredible Open Source community. This will allow students to leverage free, publicly available tools to build applications with complexity and real-world application beyond what they would ever to approach otherwise.

3- **Career Flexibility-** The ruby language allows students to explore abstract programming frameworks via a low-level, object-oriented language. As such, students are able to gain immediately relevant career skills, while gaining a foundation in a language that will afford them the ability to easily expand on their knowledge base into other relevant skillsets. Many students leverage the foundation they learned in ruby to accept full time positions that leverage other languages ranging from python and javascript to objective-c.

// FLATIRON SCHOOL

CURRICULUM OVERVIEW

Prework

Before in-person instruction begins, students complete up to 150 hours of preparatory work. The goal of the prework is to get everyone in the program to a minimum baseline, while ensuring that all admitted students are dedicated to doing the work necessary to be successful during the remainder of the course. During this portion of the program, students are not required to work on campus, though they will have the option to do so and have full access to instructors to help guide them through the work. The prework curriculum is outlined at http://prework.flatironschool.com.

Unit One – Ruby

Description: Students begin exploring version control using git commands including with cloning, branching, merging, rolling back commits, forking, and submitting pull requests. We also learn explore the history of software development from Waterfall to Version control and discuss the historical significance of open sourced tools like Linux.

Students then learn fundamental concepts in programming including repls, methods, loops, variables, variable scope, conditionals, blocks and iterators, case statements, arrays, scope, hashes, regular expressions, iterators, enumerables, data structures, nesting, etc.... Topics build in complexity and provide the foundation for the rest of the course.

Students learn to embrace error messages as clues and gain a fundamental appreciation for failure as the only way to learn and progress. They gain experience in debugging with various gems and tools designed to track down issues in code. Labs during this semester guide students in computational thinking.

Deliverable: A solid foundation in Ruby, continued experience with git, experience reading and understanding unit tests, enhanced problem-solving skills.

Unit Two – Object Orientation

Description: Students gain experience with Object Oriented Programming and understand how it allows programmers to bundle code and create reusable objects and methods, allowing for increasing complexity in software.

Deliverable: Understanding of OOP and how to implement it in Ruby.

// FLATIRON SCHOOL

Unit Three – Object Relational Mapping

Description: ORM (Object Relational Mapping) allows programmers to query and manipulate data from a database using an object-paradigm. Students learn to write manipulate data using the Ruby language. We gain an appreciation for the structure of a database, how to map out tables, and the difference between the various table relationships. Students learn how to wireframe database structures, as well as how to link their applications to a database. This unit covers sql, domain modeling, relational database theory, schema architecture, and the Object Relational Model, including the ActiveRecord pattern.

Deliverable: Comprehension of database modeling and design, understanding of what an ORM is and it's powers, ability to integrate a SQL database with an application.

Unit Four –Rack

Description: This unit is designed to give students an understanding of the Hyper Text Transfer Protocol (HTTP), and how the Internet works, as implemented through the Ruby web interface of Rack. Students build their own HTTP Servers and learn how the request / response model of the web works. Their servers listen to HTTP requests and respond with well-formed HTML responses. They learn to understand the web with the few abstractions provided by the tool set.

Deliverable: A Rack powered Ruby web application that integrates with models from the ORM unit to serve dynamically rendered HTML templates with data from a database.

Unit Five – Sinatra

Description: Sinatra is a Domain Specific Language (DSL) written in Ruby for building web applications on top of Rack. This framework provides students with exposure to design patterns in web applications. The topics covered in this unit include architectural patterns such as REST (Representational State Transfer), MVC (Model-View-Controller), HTML Forms, ERB (Embedded Ruby) and template rendering, and application environments.

Deliverable: A Sinatra powered Ruby web application that processes data from HTML forms and communicates with a model layer to communicate with the database.

Unit Six – Rails

Having a foundation in the Ruby language as well as the architecture of the World Wide Web, students use Rails to build complex functional web applications from the ground up. They learn the file structure of Rails, how to set up their own databases, how to draw

routes and create Rails forms, gain an understanding of the asset pipeline, and bring it together by integrating front-end design skills.

Students also have the ability to take on more advanced concepts such as authorization, validation and callbacks. Once students grasp the basic functionality of Rails, they spend time building out their own Rails applications, moving through the entire process from idea to execution

Deliverable: A multi-model Ruby on Rails powered web application that integrates nested forms and URL routes to a multi-relational data set.

Unit Seven – JavaScript

Javascript powers the user experience of the web. Students learn the basics of the javascript syntax, it's functional architecture, and different approaches to the object model. They then learn the Document Object Model (DOM) Javascript API provided by the browser to dynamically interact with HTML. This unit focuses on jQuery, the most popular javascript library, to aid students in learning how to collect user input, manipulate the DOM with animations and injection, and send Asynchronous AJAX requests for a rich user experience. Students then explore popular javascript frameworks including AngularJS, Ember, etc…

Deliverable: A Rails application that includes examples of DOM Manipulation, AJAX HTML Loading, and AJAX Form Submissions.

Unit Eight - Projects

Learn by doing. Every student works on 3-5 applications throughout the semester. One application will be a robust “Capstone” project, built in a team of four students over several weeks. Students will also work on smaller applications both individually and in teams. Towards the end of this period, students will have the opportunity to explore specific technologies that interest them so that they can learn more about technologies for specific industries (ie. students interested in commerce may build applications using shopping carts and payment provider integrations).

Throughout – Being a Software Developer

While the linear progression of the curriculum is focused on technical skill attainment, the course is designed to teach students how to be software developers. This includes things ranging from honing communication skills (all students maintain active technical blogs and present at technical meetups) to being comfortable learning new skills, represented in activities ranging from specific assignments to on-campus activities like lock picking, knot-tying, yoga, origami, dance, etc…