Onsite Bootcamp

PHP | MEAN | RUBY ON RAILS | PYTHON | IOS

Last Updated: 2016 July 25

TABLE OF CONTENTS

ONSITE BOOTCAMP

CURRICULUM*

Web Development
LAMP
Python
MEAN
Ruby on Rails
iOS

STRUCTURE

3 Full Stacks
14 weeks
50-80 hours/week

The Onsite Bootcamp is the most hands-on and intensive program that we offer. As a student, you will learn up to 3 of the 5 stacks from our full curriculum – LAMP, Python, MEAN, Ruby on Rails, and iOS. To start your bootcamp, you will need to choose to learn either LAMP or Python, and your two additional stacks may be chosen after you have started the program. If you're not sure which to choose, don't worry, you may submit an application and our staff will help guide you.

Students will have access to instructor support from Monday to Friday and 24/7 access to our state-of-the-art facility. Our facility includes dual monitor work stations for every student, a complimentary coffee and snack bar, a fully-equipped kitchen for meals, an immersive learning environment filled with like-minded individuals, and more. In the evenings, you will have access to remote instructor support until midnight from Sunday to Thursday and receive an account to access Coding Dojo's online learning platform, which includes volumes of video tutorials for independent studying.

Furthermore, students and alumni will have access to our Career Services program, where you'll work closely with our team to pursue short and long-term career goals. You'll be able to schedule one-on-one sessions with our Career Advisor team, attend exclusive job-hunting workshops, and more.

PREREQUISITES

- Personal laptop to work on during the program
- Pass the admissions interview
- Spent 100 hours of learning how to code on his/her own**

*The availability of stacks will vary by campus.
**Students with less than 100 hours of experience are required to complete the pre-boot camp coursework.

Week 1-2

WEB FUNDAMENTALS

Start the program by learning the fundamentals of front-end development.

WEB FUNDAMENTALS

Week 3-6

FULL STACK 1

Pick between LAMP or Python to be your first stack to learn and enter the world of back-end development.

LAMP OR **PYTHON**

Week 7-10

FULL STACK 2

Week 11-14

FULL STACK 3

Pick between MEAN, Ruby on Rails, or iOS as your second and third stacks. Our instructors will help you choose the stack best fit for you.

MEAN OR **RUBY/RAILS** OR **IOS**

Week 15-18

RESIDENCY PROGRAM

After completing the program, students may apply for the Residency Program, where alumni will be able to spend up to an extra 4 weeks on campus. During this time, participants of the program will have full access to the course material, our career support services, and mentorship from our instructors.

This is an optional program that is ideal for those who wish to strengthen their skills and utilize extra time to build their portfolios, as well as for entrepreneurs who need more time to build their product or develop their unique web application.

*The availability of stacks will vary by campus.

CURRICULUM OVERVIEW

	TECHNOLOGIES	TOPICS COVERED
WEB FUND.	HTML/HTML5 CSS/CSS3 Basic Javascript Advanced jQuery Git/Github & Terminal Responsive Web Design* Balsamiq* Bootstrap* LESS & SASS*	Front-end Development Frameworks & Libraries Wireframes & Mockups Code Version Control HTTP Request Dynamic Content
PYTHON	Python MySQL Flask Ajax APIs jQuery Django* PostgreSQL*	OOP in Python SQL Queries & ERD Diagrams Web Security Basics CRUD Operations MVC Framework & Design Patterns Application Deployment Object Relational Mapper* Web Crawler* Scaling Web Apps*
LAMP	MySQL PHP Ajax Advanced PHP CodeIgniter API Basic Javascript jQuery	OOP in PHP Web Security Basics Semi-Restful Routes SQL Querries & ERD Diagrams MVC Framework & Design Patterns CRUD Operations Application Deployment Web Crawler* Scaling Web Apps*

*Optional topics available

MEAN

TECHNOLOGIES

MongoDB
Express
AngularJS
Node.js
Advanced JavaScript
Node Package Manager
Socket.IO
Bower*

TOPICS COVERED

OOP in Javascript
Ajax Requests
Building an MVC Framework
Creating Custom JS Libraries
Building Real-time Apps
NoSQL Database Design
RESTful Routing
Agile Development*

RUBY/RAILS

Ruby
Rails
RSpec
Capybara
PostgresSQL
Active Record
Angular on Rails*
Ember*
Coffeescript*

OOP in Ruby
Object Relational Mapper
Test Driven Development
RESTful Routes
MVC Framework & Design Patterns
Authentication/Authorization
Rails Deployment
Ember + Sockets*
SASS & HAML*

IOS

Swift
Xcode
Core Data
AV Foundation
iOS Fundamentals
Core Motion
Core Location
iOS Sockets

Strongly Typed Language
OOP in Swift
User Interface Views
iOS with a Server
iOS and Sockets
Data Persistence with Swift
Storyboarding in iOS

*Optional topics available

WEB FUND.

HTML/HTML5

Intro to HTML
- Basic Nesting Practices
- The Header & Body
- Common Body Tags (lists, tables, etc.)
- Building Forms & Declaring Input Values
- Containers, Elements, Attributes, & Classes
- HTML Best Practices
- Intro to HTML5

CSS

Intro to CSS
- CSS Selectors & Declarations
- Inspecting Element
- Inline, Block, Float, and Positioning
- Div Layout & Formatting
- Styling Text & How Fonts Work
- Using Properties & Backgrounds
- Replicating Complete User Interfaces
- Optimizing & Cleaning Your Code

Intro to CSS3 & More Styling Properties
- How to Build Your Own Shapes*
- Constructing Complex Tables*
- Intro to Bootstrap*
- CSS Preprocessors, LESS, & SASS*
- Optional Frameworks, UI Assets, & Tools*

JQUERY

Intro to jQuery
- jQuery Functions & Debugging
- How to Use Parameters & Getters/Setters
- Essentials of the jQuery Library
- Troubleshooting jQuery

Intro Advanced jQuery
- Implementing Dynamic Content
- Callbacks in jQuery
- Transversing DOM Elements
- Using Forms in jQuery
- Using jQuery UI Library*
- Extra jQuery Libraries*

GIT/GITHUB

Intro to Git & Version Control
- Using Terminal Commands
- How to Create & Utilize a Repository
- Making, Tracking, & Reverting Changes
- Git Workflow Overview & States
- Advanced Git Commands & Concepts
- Branching, Merging, & Conflicts

Intro to Github
- How to Use a Github Repository
- Forking, Cloning, & Pulling
- Github Collaboration & Workflow

RWD

Intro to Responsive Web Design (RWD)
- Breakpoints, Units, & Media Queries
- Basics to Typesetting & Scaling
- Cross-device RWD
- Grid System, Fluid Grids, & Adaptive Layouts

Intro to CSS Frameworks
- Responsive Typography
- Using CSS Reset & Boilerpoint

WIREFRAMES*

Intro to Wireframes
- Importance of Wireframes
- Intro to Balsamiq & How to Use It

*Optional topics available

PYTHON

MYSQL

Intro to MySQL

- Database Design & Relationships
- Entity Relationship Diagrams (ERD)
- Database Normalization
- Intro to MySQL Workbench & Querying
- Conventions & Common Data Types
- How to Use ERDs
- Using a Database with Your UI
- Recreating ERDs*

PYTHON

Intro to Python

- Creating Variables in Python
- Common Data Types & Best Practices
- Using Strings & Built-in String Functions
- List Creation & Manipulation
- Using Tuples & Built-in Tuple Functions
- How to Use Dictionaries in Python
- Conditionals, Operators, & Nested Loops
- Constructing Functions in Python

PYTHON OOP

Python Object Oriented Programming (OOP)

- Creating Objects & Classes
- Adding Properties/Attributes to Classes
- Constructing & Adding Methods to Classes
- Chaining Methods & Using Magic Methods
- How to Use Modules & Packages in Python
- Creating Multiple Objects
- Updating Methods with ‘Super’

Intro to Python Advanced Topics

- How to Use Multiple Arguments
- Ternary Operators in Python
- Using Lambda
- Overriding Inheritance & Polymorphism
- Using Composition Over Inheritance

Python Test Driven Deployment (TDD)

- Unit Testing in Python & Outcomes
- How to Use Assertions
- Using TDD Methods: setUp & tearDown

FLASK

Intro to Flask

- Routing in Flask Applications
- Building & Using Forms
- Rendering Templates & Views
- Delivering Static Content
- The Different HTTP Methods
- Implementing Cookies & Sessions
- Hidden Inputs
- Form Validation

Intro to Flask with MySQL

- Import, Export, & Connect Your Database
- Connecting & Running Python Across Files
- Database Communication with Python
- Data Validation with Python
- Encryption & Data Security Basics
- Using BCrypt for Encryption

PYLOT MVC

Intro to Pylot Model View Controller (MVC)

- What is an MVC?
- How Controllers Work
- Rendering Views
- Session Classes & Using Session Data
- Routing in Pylot
- How to Use Models with Controllers
- Data Validation with Pylot
- Using Bcrypt with Pylot MVC
- How to Use Multiple Controllers & Models

DEPLOYMENT

Intro to Python Application Deployment

- Tools You’ll Use:
  - Amazon Web Services (EC2)
  - Linux (Ubuntu)
  - Gunicorn & Nginx
  - PostgreSQL
  - Virtualenv
  - Git
  - Custom Domains

*Optional topics available

LAMP

MYSQL

Intro to MySQL

Database Design & Relationships
Using Entity Relationship Diagrams (ERD)
Database Normalization
Intro to MySQL Workbench & Querying
Conventions & Common Data Types

PHP

Intro to PHP Fundamentals

Declaring Variables & Array Variables
Conditionals, Operators, & Nested Loops
Array Manipulation
Associative & Multidimensional Arrays
Utilizing Built-in Functions
How to Construct Functions & Debugging

PHP Advanced Topics

Data Transfers & Get/Post
Implementing Cookies & Sessions
How to Utilize Headers with PHP
Hidden Inputs

Integrating PHP with MySQL

Import, Export, & Connect Your Database
Database Communication with PHP
Data Validation with PHP
Blocking MySQL Injections
Encryption & Data Security Basics

CODEIGNITER

Intro to CodeIgniter Fundamentals

What is an MVC?
How Controllers Work
Views & Passing Data
Using Input Classes & Security
Session Classes & Using Session Data
Using Models with the Database/Controllers
Data Validation with CodeIgniter
How to Use Multiple Controllers & Models

CodeIgniter Advanced Topics

Client-side Validation, Functions, & Listeners
Client-side & Controller Validation
Client-side & Model Validation
Redirecting vs. Calling a Controller

PHP OOP

PHP Object Oriented Programming (OOP)

Creating Objects & Classes
Adding Properties/Attributes to Classes
Constructing & Adding Methods to Classes
Creating Multiple Objects
Magic Methods & Instantiation
How to Chain Methods

OOP Advanced Topics

Procedural Programming vs. OOP
Extending & Inheriting Classes
Overwriting or Preserving Parent Classes
Visible, Private, & Protected Inheritance
Intro to Linked Lists & Data Structures
Singly & Doubly Linked Lists

AJAX

Ajax API with jQuery

Intro to Application Programming Interfaces
Requesting from APIs
Using JavaScript Objects (JSON)

Ajax API with jQuery & CodeIgniter

How Web Applications Work & HTTP Request
API with JSON & HTML
Debugging JavaScript & jQuery
Access Control Origin
Connecting to Various APIs
Construct an Ajax Enabled App

DEPLOYMENT

Intro to PHP Application Deployment

Tools You'll Use:
- Heroku
- Azure
- Hostmonster
- Amazon Web Services (EC2)
- Rackspace
- GoDaddy

*Optional topics available

MEAN

JAVASCRIPT

Intro to JavaScript Fundamentals (ES5 & ES6)

- Declaring & Referencing Variables
- Variable Hoisting in JavaScript
- Conditionals, Operators, & Nested Loops
- Using Arrays & Loops in JavaScript
- Objects, Functions, & Function Scoping
- Variable Hoisting with Scoping
- Return Statements in JavaScript
- Function Hoisting

JavaScript Object Oriented Programming (OOP)

- How to Use Object Constructors
- Common Constructors: 'This' & 'New'
- Private Methods & Variables
- Creating Prototype Objects in JavaScript
- Best Practices for JavaScript OOP

Intro to JavaScript Advanced Topics

- How to Use Callbacks
- Delegating Functionality & Event Handling

NODE.JS

Intro to Node.JS

- How to Use Package Managers (NPM/Bower)
- File System Module & HTTP
- Making a Full Web Sever
- How to Work with Node Modules
- Common & Useful Node Modules

Modularization

- Using Require & Module.exports
- How to Modularize Existing Projects

EXPRESS.JS

Intro to Express.JS

- Render Templates With Express View Engines
- HTTP Methods: Forms, Data Tranfers, & Routing

Intro to Socket.io

- Applications with Real-time Communication

MONGO DB

Intro MongoDB

- CRUD Operations for MongoDB

Intro to Mongoose

- Dependencies in Mongoose
- Mongoose Communication with MongoDB
- Mongoose Methods
- Data Validation with Mongoose
- Create Associations Between Mongo Objects
- RESTful Routing with Mongoose & Express

ANGULAR.JS

Intro to Angular.JS

- Dependencies for Angular
- Directives, Data Binding, & Compiling
- Using Modules in Angular
- Controllers, $scope, & 'this'
- How to Create Factories
- Using Data Filters in Angular
- Ajax Requests Using Angular

MEAN

Building MEAN Applications

- Connecting Angular to Node
- Making API Requests in MEAN
- Tracing Data in the MEAN Stack

DEPLOYMENT

Intro to MEAN Application Deployment

- Tools You'll Use:
  - Heroku
  - Amazon Web Services (EC2)
  - Linux Servers

*Optional topics available

RUBY/RAILS

RUBY

Intro to Ruby Fundamentals
- The Elegance of Ruby
- Using Puts, Strings, & Basic Ruby Syntax
- Conditional Statements in Ruby
- For Loops & Arrays in Ruby
- How to Use Iterators & Blocks
- Intro to Modules in Ruby & Enumerable

Intro to Ruby OOP
- Creating Classes, Methods, & Properties
- Using Private Methods
- Working with Inheritance in Ruby

Ruby Test Driven Deployment (TDD)
- Intro to RSpec Methods
- How to Write Tests in TDD
- Implementing Test Driven Deployment

RAILS PART 1

Intro to Rails Model View Controllers (MVC)
- How to Get Started with an MVC
- Intro to Gems
- Using Models in Rails
- How to Use ORM in Rails
- Validations, Relationships, & Migrations

Intro to Controllers & Views
- Using Restful Routes & Routing in Rails
- How to Use Controllers
- Passing Information with Variables
- Rendering Data with Controllers
- How to Use Views
- Intro to Form Helpers
- Using Scaffolding in Your Projects
- Basic Web Security in Rails Part 1
- Patch & Delete Methods in Restful Routes
- Basic Web Security in Rails Part 2
- TDD vs. Error Driven Development (EDD)
- Layouts with Controllers & Views

TDD

Intro to RSpec & Capybara
- Using Expectations in RSpec
- Using “Describe” & “It”
- Testing Your Models with RSpec
- RSpec with Capybara Part 1
- RSpec with Capybara Part 2
- How to Test Routes in RSpec
- Testing with RSpec in Various Scenarios

RAILS PART 2

Intro to TDD in Rails
- User Permissions in Rails
- Intro to Postgres & Database Setup
- Basic Encryption in Rails
- User Authentication in Rails
- User Authorization in Rails
- How to Build App Features with Rails & TDD

RAILS PART 3

Intro to Ajax, Gems, OAuth, & APIs in Rails
- How to Use The Asset Pipeline
- Using Ajax with the Rails Framework
- API Integration in Rails
- OAuth, Graph API, & REST API
- Uploading Files with Paperclip
- Integrate Rails with Node.js & Express.js

DEPLOYMENT

Intro to Rails Application Deployment
- Tools You’ll Use:
  - Heroku
  - Amazon EC2

ANGULAR ON RAILS

Intro to Ruby on Rails with Angular
- Using Angular Route Libraries
- How to Create Models with Rails & Angular

*Optional topics available

IOS

SWIFT BASICS

Intro to Swift Fundamentals

- Data Types: Constants & Variables
- Conditional Statements, Operators, & Loops
- Basic Types & Typecasting
- Array Manipulation
- How to Use Dictionaries in Swift
- Swift Optionals
- Swift Object Oriented Programming (OOP)
- Creating Classes & Structs
- Inheritance in Swift
- Value vs Reference Types
- Using Functions in Swift

IOS BASICS

Intro to iOS Fundamentals

- Storyboarding in iOS
- Working with Autolayout
- Linking your Storyboard to Code
- Intro to Xcode
- How to Use a Debugger
- View Lifecycle Basics

IOS INTERMEDIATE

Intro to iOS Intermediate Topics

- Using CoreData
- Storing User Defaults in iOS
- Using Protocols & Delegates
- How to Use Table Views
- Collection Views
- Segueing Between Views
- Using Navigation
- Tab Bar Controllers

IOS ADVANCED

Intro to iOS Advanced Topics

- Linking iOS to a Back-end Server
- Making HTTP Requests in iOS
- JSON Data in Swift
- Grand Central Dispatch in iOS
- Using Type Coercion

*Optional topics available

ADMISSIONS PROCESS

1. APPLICATION

Prospective students must first submit an admissions application. This is a brief application form where you'll share your background, submit your resume, and provide contact information. Don't worry, we aren't specifically looking for coding experience. This is simply a chance for us to learn more about you.

2. ADMISSIONS ORIENTATION

The next step is to complete our optional Admissions Orientation. This is a brief 6-min walk through about who we are, who this program is for, and what to expect as a Coding Dojo student.

3. INTERVIEW

In this step, we'll take this time to see if you're a good fit for the program. We firmly believe that we can teach anyone how to code, however we also need to ensure our students are prepared for the challenges of the boot camp. This interview and your application will be factored into our admissions decision, which will be made 3-5 business days after your interview.

4. ACCEPTANCE LETTER

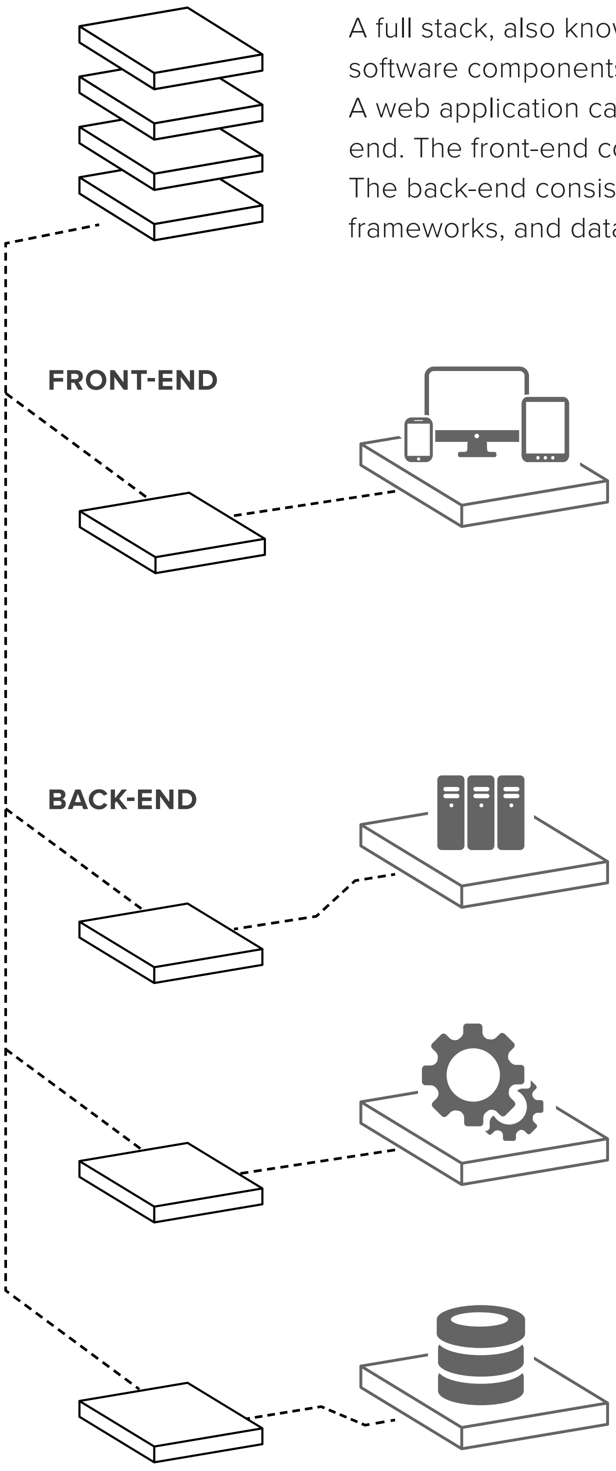
If selected to attend, you will receive an acceptance letter through email and a link to submit your safety deposit, which will reserve your seat in the Coding Dojo program. You will also receive instructions concerning the required preparations for your upcoming program.

5. SAFETY DEPOSIT

Due to limited seats and high demand, you must first submit your safety deposit to reserve your seat and access the pre-course materials.

WHAT IS A FULL STACK?

A full stack, also known as a software stack or bundle, is a set of software components needed to create a complete web application. A web application can be divided into two areas: front-end and back-end. The front-end contains client-side languages and frameworks. The back-end consists of web servers, back-end languages/frameworks, and databases.



CLIENT-SIDE LANGUAGES/FRAMEWORKS

HTML*
CSS
JAVASCRIPT
ANGULAR
JQUERY

WEB SERVERS

APACHE*
NGINX
IIS
NODE.JS

BACK-END LANGUAGES/FRAMEWORKS

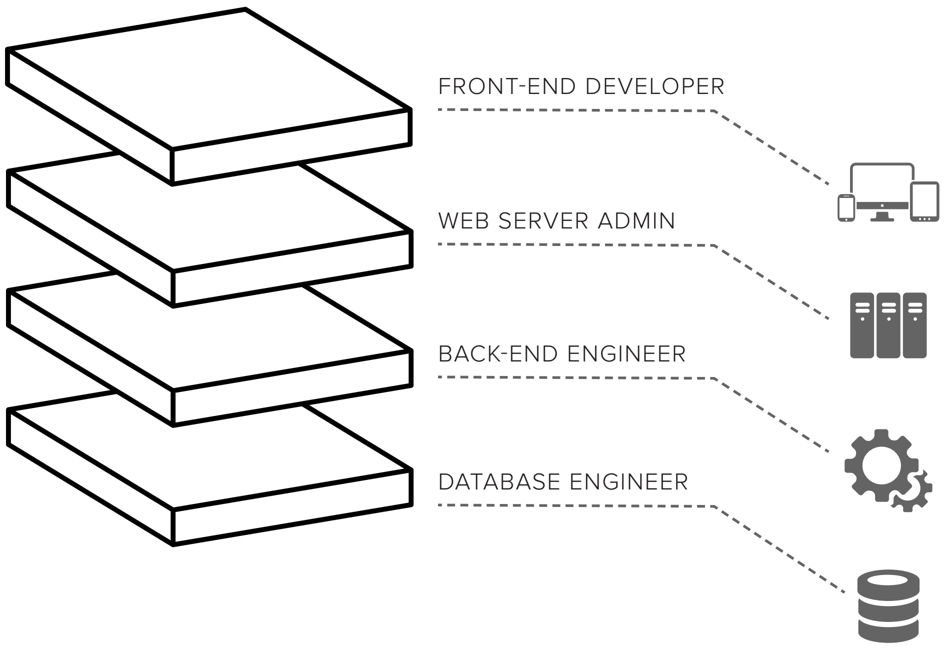
PHP/CODEIGNITER*
RUBY/RAILS
PYTHON/DJANGO

DATABASES

MYSQL*
MONGODB
REDIS

*Popular languages and technologies.

FULL STACK DEVELOPER



Full Stack Developers are well-rounded software engineers who have the know-how to independently build fully functional platforms, from the front-end to the back-end. Conventionally, web development requires several variations of engineers: front-end developers, web server administrators, back-end engineers, and database engineers. However a full stack developer is all of the above, and whether in a large or small engineering team, s/he can add value and insight to all layers of the project.