Master Course Packet
Onsite & Online
Full Time & Part-Time
Full Stack Curriculum

5000+ grads to date
$60k - $125k avg alumni salary*

Over 5000 alumni hired by tech companies worldwide

*As of Feb 2018 alumni data
Onsite Bootcamp
Currently offered online due to COVID-19

Your career as a software developer starts on your first day in class.

Within 14 weeks we’ll turn you into a self-sufficient, versatile developer who has all the critical skills to have a long, healthy career in tech.

Learn by Doing
You’ll start coding from day one on campus. Dive into a fast, project-based learning environment that fosters collaboration, not competition.

Anyone Can Learn to Code
Anyone can learn to code, but the path to becoming a developer isn’t easy. The most successful students dedicate at least 70-90 hours/week to the bootcamp.

A Typical Day at the Dojo

Activities subject to change based on campus and curriculum
3 Full Stack Curriculum

We’re here to maximize your career opportunities and coding mastery. You’ll learn 3 full stacks, have a portfolio to show, and 3x the job prospects.

Level Up, Stack by Stack

Web Fund.
- Terminal
- Git/GitHub
- HTML5
- CSS3
- Javascript
- jQuery

Python
- Python 3
- OOP
- Flask
- Django
- MySQL
- Ajax

C#.NET
- C#
- ASP.NET Core 2
- LINQ
- Dapper
- Entity Framework
- Identity

Python
- Python 3
- OOP
- Flask
- Django
- MySQL
- Ajax

C#.NET
- C#
- ASP.NET Core 2
- LINQ
- Dapper
- Entity Framework
- Identity

Ruby on Rails
- Ruby
- Rails
- RSpec
- Capybara
- PostgreSQL
- Active Record

Java
- Java 8
- MySQL
- JSPs
- Spring Data JPA
- Spring Boot
- Spring Security

MERN
- Javascript ES6
- MongoDB
- Express.js
- React
- Node.js
- Socket.io

Awards & Recognition

Curriculum subject to change during attendance due to mid-course improvements
Online Part-Time

In 16 to 28 weeks, you can transition to a career in development without quitting your day job.

This program is a flexible alternative that provides full, online access to our Python curriculum -- complete with live support and collaboration with instructors and classmates.

Two Options to Fit Your Schedule

<table>
<thead>
<tr>
<th>ACCELERATED</th>
<th>FLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>16</strong> weeks</td>
<td><strong>28</strong> weeks</td>
</tr>
<tr>
<td><strong>25</strong> hrs/wk</td>
<td><strong>14</strong> hrs/wk</td>
</tr>
</tbody>
</table>

Complete web fundamentals, then choose from the following stacks:

Complete web fundamentals, then start Python

ONLY Python is available through Flex at this time.
ACCELERATED

Learn to build applications in the top programming stacks of 2020. Pick between Python, MERN, or Java as your stack, or choose to extend the program and learn multiple languages.

Your Progression Plan

Week 1 - 4
Web Fundamentals
- HTML
- CSS
- JavaScript

Week 5 - 12
Pick Your Full Stack
- PYTHON
  - Python
  - OOP
  - Django
  - MySQL
  - Ajax
- MERN
  - JavaScript
  - MongoDB
  - Express
  - React
  - Node
  - Socket.IO
- JAVA
  - Java
  - MySQL
  - JSPs
  - Spring Data
  - Spring Boot
  - Spring Security

Week 13 - 16
Projects & Algos
Projects
Algorithms

Weeks 17+
Optional: Add More Stacks
This is a great option for students that want to take the full-time curriculum over a longer time period, or if you aren't sure which stack you want to specialize in.

You can also choose to add a stack after you've started or completed your first (so you can see if you enjoy it before you commit).

Each additional stack is 8 weeks long.

A Typical Week in the Part-Time Program

1 Hr Live Lecture
5PM
Monday

1 Hr Live Lecture
5PM
Tuesday

Code Review
5PM
Wednesday

Thursday

Friday

Lectures are delivered either on Mon/Wed or Tues/Thurs
Flex program only has one lecture per week

Activities subject to change based on campus and curriculum

Self Study
20-30 hours/wk in Accelerated
10-15 hours/wk in Flex

30 min. Code Review
available for assignment feedback and help Monday-Friday as instructors’ schedule allows

TA Support
Mon-Fri: 11:00am - 8:00pm
Sat: 8:00am - 6:00pm
Sun: 8:00am - 2:00pm

All times in PST
### Time Management

Here’s what a typical week might look like for someone who continues to work full-time as well as participate in family activities while in the Accelerated program.

<table>
<thead>
<tr>
<th>SUN</th>
<th>MON</th>
<th>TUE</th>
<th>WED</th>
<th>THU</th>
<th>FRI</th>
<th>SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catchup of friends</td>
<td>10am - noon</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Assignments &amp; Check in</td>
<td>1 - 3pm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Family Time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Family Dinner</td>
<td>5 - 6 pm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prep</td>
<td></td>
<td>Prep for week; organizing home office</td>
<td>7 - 10 pm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6am - 8pm</td>
<td>6am - 8pm</td>
<td>HTML</td>
<td>6am - 8pm</td>
<td>HTML</td>
<td>6am - 8pm</td>
<td>CSS</td>
</tr>
<tr>
<td>Workout</td>
<td>Workout</td>
<td>Take care of daily work</td>
<td>6am - 8pm</td>
<td>Workout</td>
<td>Take care of daily work</td>
<td>CSS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Workout</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Family Dinner</td>
<td>5 - 6 pm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prep</td>
<td></td>
<td></td>
<td>Family Time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Family Dinner</td>
<td>5 - 6 pm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prep</td>
<td></td>
<td>Prep for week; organizing home office</td>
<td>7 - 10 pm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTML</td>
<td>HTML</td>
<td>Take care of daily work</td>
<td>6am - 8pm</td>
<td>HTML</td>
<td>Take care of daily work</td>
<td>CSS</td>
</tr>
<tr>
<td>Orientation + Lecture</td>
<td>6am - 8pm</td>
<td>6am - 8pm</td>
<td>Lecture</td>
<td>6am - 8pm</td>
<td>CSS</td>
<td>8 - 10 am</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Family Dinner</td>
<td>5 - 6 pm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prep</td>
<td></td>
<td>Prep for week; organizing home office</td>
<td>7 - 10 pm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTML</td>
<td>HTML</td>
<td>Take care of daily work</td>
<td>6am - 8pm</td>
<td>HTML</td>
<td>Take care of daily work</td>
<td>CSS</td>
</tr>
<tr>
<td>Assignments</td>
<td>Assignments</td>
<td>Take care of daily work</td>
<td>8 - 10 pm</td>
<td>Assignments</td>
<td>Work on assignments; apply lecture stuff</td>
<td>8 - 10 pm</td>
</tr>
<tr>
<td>Baby Time</td>
<td>Baby Time</td>
<td></td>
<td>Baby Time</td>
<td>Baby Time</td>
<td>Baby Time</td>
<td>Baby Time</td>
</tr>
<tr>
<td>8pm</td>
<td>8pm</td>
<td></td>
<td>8pm</td>
<td>8pm</td>
<td>8pm</td>
<td>8pm</td>
</tr>
<tr>
<td>Rest</td>
<td>Rest</td>
<td></td>
<td>Rest</td>
<td>Rest</td>
<td>Rest</td>
<td>CSS</td>
</tr>
<tr>
<td>8:30 - 10 pm</td>
<td>8:30 - 10 pm</td>
<td></td>
<td>8:30 - 10 pm</td>
<td>8:30 - 10 pm</td>
<td>8:30 - 10 pm</td>
<td>8:30 - 10 pm</td>
</tr>
</tbody>
</table>

### Pro Tips from Student Success

**Overestimate the time you need for self-study**

The Part-Time Online program expects you to dedicate at least 20 hours per week in the learning platform working through content. So, for the first few weeks, allocate 24 hrs for that work. It is easier to scale back than scale up.

**Create a calendar and stick with it!**

It sounds simple, but a calendar can be shared with family and friends to help you stay accountable and to get insight into when you’re going to be heads down. It also gives you a reality check into how much time you actually spend.

**List out responsibilities and see who can help**

Create a list of your household and family responsibilities. See if you can offload any tasks or get additional help from housemates, friends, and family. If you’ll be working during this time, do the same exercise with coworkers.
FLEX
The same Python curriculum, over a longer amount of time, so you can manage the rest of your commitments more easily.

Your Progression Plan

Week 1 - 8
Web Fundamentals
- HTML
- CSS
- JavaScript

Week 9 - 24
Python Full Stack
- Python
- OOP
- Django
- MySQL
- Ajax

Week 25 - 28
Projects & Algorithms
- Projects
- Algorithms

Unlike the Accelerated program, you do not have a choice of stack.
You also do not have the option to add any additional stacks at this time.

Whether you choose Accelerated or Flex, we are here to support you.

Hands-on, Structured Teaching
Dive into an immersive online learning environment filled with live mentorship, instruction, and collaboration with real instructors and classmates.
All from the comfort of your own home.

Anyone Can Learn to Code
Anyone can learn to code, but the path to becoming a developer isn’t easy. Students typically dedicate 20-30 hours a week to self-study in the accelerated program, and 10-15 hours in Flex.

Curriculum subject to change during attendance due to mid-course improvements
Online Full-Time

No matter where you are in the world, your career as a software developer starts on your first day.

Within 14 weeks we’ll turn you into a self-sufficient, versatile developer who has all the critical skills to have a long, healthy career in tech.

Hands-on, Structured Teaching

Dive into an immersive online learning environment filled with live mentorship, instruction, and collaboration with real instructors and classmates.

All from the comfort of your own home.

Anyone Can Learn to Code

Anyone can learn to code, but the path to becoming a developer isn’t easy. The most successful students dedicate at least 70-90 hours/week to the bootcamp.

A Typical Day in the Online Bootcamp

Activities subject to change based on campus and curriculum
3 Full Stacks Online
We’re here to maximize your career opportunities and coding mastery. You’ll learn 3 full stacks, have a portfolio to show, and 3x the job prospects.

Level Up, Stack by Stack

JAVA
Java 8
MySQL
JSPs
Spring Data JPA
Spring Boot
Spring Security

Python
Stack 1
HTML
CSS
JavaScript
Python 3
OOP
Flask
Django
MySQL
Ajax

JAVA
MERN
Stack 3

JavaScript
ES6
MongoDb
Express.js
React
Node.js
Socket.io

Curriculum subject to change during attendance due to mid-course improvements
### HTML

**Intro to HTML**
- Basic Nesting Practices, Indentation
- The Head & Body
- Body Tags (lists, tables, etc.)
- Building Forms & Declaring Input Values
- Containers, Elements, Attributes, & Classes

### 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

**Intro to CSS3 & More Styling**
- Building Shapes
- Constructing Complex Tables
- Intro to Bootstrap
- CSS Preprocessors, LESS, & SASS

### Git / Github

**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

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

### JQuery

**Intro to jQuery**
- jQuery Functions & Debugging
- Parameters & Getters/Setters
- Essentials of the jQuery Library

**Advanced jQuery**
- Implementing Dynamic Content
- Callbacks in jQuery
- Traversing DOM Elements
- Forms in jQuery
- jQuery UI Library & More Libraries

### Responsive Web Design

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

**CSS Frameworks**
- Responsive Typography
- Using CSS Reset & Boilerpoint

**Wireframing**
- Balsamic Overview
- Wireframing Fundamentals

*Optional topics*
Python
Full Stack Development

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

Python
Intro to Python
  Variables, 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
Intro to Object Oriented Programming
  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’

Python Test Driven Development (TDD)
  Unit Testing in Python & Outcomes
  How to Use Assertions Using
  TDD Methods: setUp & tearDown

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

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

Flask w/ SQL
  Import, Export, & Connect Your Database
  Connecting & Running Python Across Files
  Database Communication & Validation
  Encryption & Data Security Basics

Deployment
  Amazon Web Services (EC2)
  Linux
  PostgreSQL

*Optional topics
Java Fundamentals

Intro to Java
- Java Development Kit Installation
- Executing Java Programs
- Variables, Data Types, & Type Casting
- Control Structures & Exceptions

Java OOP
Intro to Object Oriented Programming
- Creating Objects & Classes
- Methods, Member Variables & Constructors
- Overloading & this
- Inheritance & Packages

Advanced Java OOP
- Use of Static
- Interfaces & Abstract Classes
- Annotations
- Java Beans

Data Structures*
- Doubly Linked Lists
- Tries

Java Web Development
Java on the Web
- Servlets & Web Containers
- Query Parameters
- Java Servlet Pages
- Light MVC Patterns
- Session & POST Patterns

Java Spring

Spring Fundamentals
- Spring Overview
- Spring Tool Suite
- Intro to Spring Boot
- Spring MVC Apps

Spring Data I & II
- MySQL Connections
- Repositories & Spring Data - JPA
- Persistent Model Annotations
- Relationships
- Advanced Queries

Spring Security
- Spring Security Overview
- Authentication & Authorization
- Servlet API Integration
- Spring MVC Integration

Deployment
- Amazon Web Services (EC2)
- Linux
- PostreSQL

*Optional topics
JavaScript

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

Advanced JavaScript
- How to Use Callbacks
- Delegating Functionality & Event Handling

Node.JS

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

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

Express.JS

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

Socket.io

Applications with Real-time Communication

MongoDB

MongoDB & Mongoose
- MongoDB Overview, CRUD Ops
- 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

React

Create React App
- Class Based Components
- Props, Children, Synthetic Events
- State, LifeCycle Methods
- Functional Components
- useState, useEffect, useReducer
- context API

Deployment

Amazon Web Services (EC2)
Linux
Production Environments
Heroku

*Optional topics
C# Fundamentals
Intro to C#
- .NET Core Console Applications
- Variables, Types, Type Casting, & Functions
- Control Structures
- Debugging .NET Core Applications (VS Code)

C# OOP
Intro to Object Oriented Programming
- Classes & Objects
- Access Modifiers
- Inheritance & Polymorphism
- Encapsulation with Properties

Advanced C# OOP
- Interfaces
- Abstract Classes
- Generics

Data Structures
- Singly Linked Lists
- Doubly Linked Lists
- Tries

ASP.NET Core
- Dependency Injection with ASP Services
- MVC Architecture
- Razor View Engine
- View Modeling
- Extension Methods
- Custom User Authentication/Authorization

Object Relational Mapping (ORM)
Working with ORMs
- LINQ
- Dapper
- Entity Framework Core

Identity Framework Core
- User Authentication/Authorization
- Identity Roles
- Third Party OAuth

Deployment
- Amazon Web Services (EC2)
- Linux
- Production Environments
- Hosting with Nginx/Supervisor

*Optional topics
How to Enroll

The goal of admissions is to make sure you’re ready for the challenges of the program, and that Coding Dojo is the right fit for you.

We look for motivation, how well you work with others, and perseverance.

1. Submit Application
2. Schedule Interview
3. Admissions Decision
4. Deposit to Enroll

Financing

Standard
- 2 Payments
- 50% of tuition due week 1
- 50% of tuition due midway

Monthly Plan
- As low as $250/month
- 30/60 month terms available

Financing provided by: SkillsFund

Top Resources

Visit Our Campus
  Link >>

Start Application
  Link >>

See Start Dates
  Link >>