PDX Code Guild

Fall/Winter 2014 Full Time Day 12-Week

Python-Based Junior Developer Bootcamp

Course Syllabus

Instructor	**Christopher Jones**	**Term**	**March 27th to July 21st**
Phone	(503) 277-9710	**Days**	Monday -Friday
E-mail	chris@pdxcodeguild.com	**Class Hours**	6:00pm - 9:30pm
Office Hours	By appointment	**Clock Hours**	280

Description:

Python-based Junior Developer is an intense twelve-week, part time, hands-on immersive course that combines individual projects and group study to give students the skills and habits they need to succeed as a junior developer.

Overview:

Students will learn Python, Django, JavaScript, SQL, HTML, CSS, how to think like a programmer, and important developer practices including source control, testing, and debugging. Students practice skills using pair programming and group work, as well as work on personal portfolio projects.

Prerequisites:

Student must be comfortable working on a computer, be able to launch applications, use a text editor, browse the Internet and install software using an install wizard.

Objectives:

The course is broken down into five sections. Each section has a number of competencies that will be incorporated into exercises and projects for each section. The sections with Standards and competencies are as follows:

Section 1:

- Python: Read, write and debug programs in Python, using professional tools and practices that meet industry expectations of a junior web developer and follow PEP 8 standards.
- Write a python program using PEP 8 standards (Industry standards for Python)
- Use comments to clearly explain code
- Use command line, IDE/text editor, python packages and repositories.
- Be professional while pair programming. (Be courteous, be flexible, communicate clearly, listen carefully and be an active participant)

Section 2:

- HTML and CSS: Create a static website using HTML5 and CSS3 that met the industry standards of W3C.
- Create a static website using HTML5 and CSS3
- Practice website file management
- Deploy website to host
- Find and correct errors in HTML and CSS

Section 3:

- JavaScript: Read and write functional JavaScript. (There is no industry standard for JavaScript)
- Augment static website using JavaScript
- Use the JavaScript libraries JQuery

Section 4:

- Framework: Use the Python web framework Django with Python, HTML, CSS, SQL and JavaScript to create a fully functional, modern website.
- Use Python HTML, CSS, SQL and JavaScript to create a fully featured website.
- Write code that uses the application-database relationship and common database components including data types, tables and stored procedures
- Independently find answers to technical questions.

Section 5:

- Capstone: Plan, design and implement a final project that demonstrates an understanding of all the topics covered, and how they work together for full stack web design. Project will follow all industry standards for the languages and tools used.
- Choose a final project that uses tools learned in class
- Break a problem down into steps and to order the steps logically
- Utilize resources to find answers to questions that come up during building of the final project
- Work independently and as a team to manage time, communicate, be flexible and delegate. Be professional, courteous and responsible.
- Produce a final project that demonstrates mastery of programming skills and professional developer practices

Texts and Materials Used in this Course:

Required: Student must bring own laptop.

- **Minimum System Requirements:**
  - **Processor:** Any recent Intel or AMD processor should do.
  - **Memory:** You will need at least 512 MB of RAM (the more the better)
  - **Hard disk space:** You will need at least 10 Gigs of hard disk space.
  - **Supported Operating systems:** Windows (XP and later), most Linux distributions, Mac OS.

References used in class:

Note: students are not required to purchase any of the references used in this course.

- Learn Python the Hard Way by Zed Shaw, HTML (online) version
  http://learnpythonthehardway.org/book/
- Python Programming: an interdiction to computer science. Second Edition
  http://mcsp.wartburg.edu/zelle/python/
- HTML & CSS by Jon Duckett
  http://www.htmlandcssbook.com/
- HTML Dog, HTML Beginner Tutorial
  http://htmldog.com/guides/html/beginner/
- Mozilla Developer Network, Getting Started with CSS
  https://developer.mozilla.org/enUS/docs/Web/Guide/CSS/Getting_started
- JAVASCRIPT & JQUERY by Jon Duckett
  http://www.javascriptbook.com/
- SQL Course.com – Interactive Online SQL Training
  http://www.sqlcourse.com/index.html
- Eloquent JavaScript: A Modern Introduction to Programming by Marijn Haverbeke.
  http://eloquentjavascript.net/
- Django Unleashed by Andrew Pinkham
  https://django-unleashed.com/
- How to Tango With Django by Leif Azzopardi and David Maxwell
  http://www.tangowithdjango.com/book
- Official Django Tutorials from Dajngoproject.com
  https://docs.djangoproject.com/en/1.5/intro/tutorial01/

Basis for Final Grade

Grading is on a Pass with Distinction/Pass/Fail scale. Students are graded on each individual section based on specific in-class exercises and/or projects and tech challenges given at the end of each unit.

In-class exercises: Students edit and debug each exercise or project until it passes. All exercises and projects submitted for grading must pass for the student to pass the course. To receive a grade of Pass with Distinction, student must do more than the minimum required on exercises, successfully using additional skills outside the lesson, for exercises and work on their project.

Pair Programming: A portion of your grade in this section will be based on your interpersonal skills while pair-programming. You must pass pair-programming to pass the class.

Tech challenges: At the end of each unit, students will be given one or more problem to work on independently and one interactive challenge/code review. Tech challenges will be graded on a curve. Students must pass tech challenges in four of the five sections to pass the course.

To achieve grade of **Pass with Distinction** for the course, student must:

- Pass pair programming,
- Pass each exercise, with an overall grade of pass with distinction in four of the five sections
- Pass each tech challenge, and interactive challenge/code review, receiving a grade of pass with distinction on at least three of them.

Course Expectations

Timely work policy:

Students work on projects in class with the support of the instructor, other students, documentation and online resources. Students work on projects until they pass. Students are expected to keep up with the rest of the class. Students unable to keep up with the class will be asked to have a meeting with the instructor and director to develop a plan to get student caught up. If student fails to get back on track within two weeks, student may be asked to take a leave of absence until the student's situation is changed sufficiently to allow student to keep up with the class.

Attendance Policy:

PDX Code Guild maintains attendance records for each student. Students are expected to be on time and attend all scheduled class times. The school requires ninety percent (90%)

completion of class hours in order to receive a certificate of completion from the course. If in any fourteen-day period your attendance is less than 90%, you will be notified and placed on probation for a period of fourteen days. If you meet the attendance requirement in the next fourteen days, you will be removed from probation. If you fail to correct your attendance problem, you will be dismissed from the school.

If dismissed from the school, you will be eligible for re-admittance without filling out a new application after a minimum period of 60 days. You may be required to provide proof that the problem that caused your chronic absenteeism has been resolved.

Conduct:

Students are expected to comply with the PDX Code Guild Code of Conduct. Students are given a copy of the PDX Code Guild code of conduct and a copy of the school catalog containing the Code of Conduct and Policy on Code of Conduct Infringement upon registration. Please refer to your copy of the code of conduct and Policy on Code of Conduct Infringement sections of the school catalog for more information.

Plagiarism:

All work submitted must be the student's own work. It is acceptable and expected that students will use online and print resources and work with classmates to complete assignments. It's normal in programming to use bits of code that someone else has written. Be careful when using bits of code written by someone else; you must make sure that you understand what each line of code does. Give credit when you work with others or use parts of existing code.

It is unacceptable to copy someone else's work in its entirety and submit it as your own work.

Course Schedule:

Each week will consist of workshop days or lab days. Workshops are led by an instructor and consist of short lectures and hands-on practice. Labs may be led by a teaching assistant and consist of hands-on practice. It's very important that you make the most of both workshop and lab so that you can keep up with the class.

Classes often take more time in one section or another so time is built in at the end of the course to compensate. Usually you have 3 weeks of capstone work time.

Holidays

- Thursday, May 18th - Pycon
- Friday, May 19th - Pycon
- Monday, May 29th - Memorial Day
- Monday, July 3rd - Independence Day
- Tuesday, July 4th - Independence Day

Week 1

- Python
    - Setting up your environment
    - Command Line Interface
    - Variables
    - Basic Variable Naming
    - Values and Types
    - Operators and Expressions
    - Evaluation Order (PEMDAS)
    - Comparison Operators
    - Basic Math
    - Imperative Programming (give computer commands)
    - Basic Casting
    - Functions
    - Scope
    - Function Calling
    - Main Function
    - Function Naming
    - Comments
    - IDE's
    - Basic Debugging
    - Source Control

Week 2

- Python
    - Basic Git
    - Booleans
    - Branching and Blocks
    - Calling Type Methods
    - Optional Arguments
    - Basic String Methods
    - Standard Library Docs
    - Indexing
    - Slicing
    - Git Branching and Merging
    - Docstrings
    - Functions as Structure
    - Stubbing
    - Refactor
    - Lists
    - List Comprehensions
    - Tuples
    - Pip

- Advanced String Methods

Week 3

- Python
  - Searching for Solutions
  - Advanced Debugging
  - File IO
  - Dictionaries
  - Dictionary Comprehensions
  - For Loops
  - While Loops

Week 4

- Python
  - Basic Objects and Classes
  - Class __init__
  - Data Design
  - Class String Representation
  - Encapsulation
  - Debuggers and PyCharm
  - Modules
  - Testing

Week 5

- HTML & CSS
  - High-Level Structure of a Web App
  - Front-End vs Back-End
  - Capstone Introduction
  - Capstone Proposal
  - Front-End Tools
  - Interface Principles
  - HTML Basics
  - Semantic Elements
  - Hierarchy
  - Element Reference
  - Web Inspector Structure Basics
  - Form Basics
  - Form Element Reference
  - Classes and IDs
  - CSS Selectors
  - CSS Values
  - Style Locations
  - CSS Patterns
  - Web Inspector Style Basics
  - Default Layout
  - Hierarchical Layout
  - Float

    - Flexbox
    - Layout Problem Solving

Week 6

- HTML & CSS
    - Form Basics
    - Form Element Reference
    - Classes and IDs
    - CSS Selectors
    - CSS Values
    - Style Locations
    - CSS Patterns
    - Web Inspector Style Basics
    - Default Layout
    - Hierarchical Layout
    - Float
    - Flexbox
    - Layout Problem Solving

Week 7

- JavaScript
    - JavaScript Basics
    - Basic Types and Variables
    - Basic Objects and Arrays
    - Basic Operators
    - Control Statements
    - Functions
    - Errors
    - Casting
    - Global Objects
    - Calling Prototype Methods
    - Basic Strings
    - Anonymous Functions

Week 8

- JavaScript
    - Iterating
    - For Loops
    - Data Transformations
    - Debug Output
    - Style Guide
    - JS in the Browser
    - Web Inspector Debugger
    - JS Debugging
    - Object Methods
    - Constructors
    - Prototypes

Week 9

- JavaScript
    - Prototype Methods
    - JS Objects Review
    - Basic Document Object Model
    - jQuery
    - DOM Queries
    - Element Creation
    - DOM Manipulation
    - Events and Callbacks
    - Front-End Problem Solving
    - JSON
    - APIs
    - HTTP APIs

Week 10

- Django
    - Dependencies
    - Virtual Environments
    - App Requirements
    - App Structure
    - Web Basics
    - URLs
    - Django Overview
    - Web Framework
    - Views
    - Routes
    - Regular Expressions
    - Parameters
    - Templates
    - Object Relational Mapper (ORM)

Week 11

- Django
    - Building an API
    - Custom template tags
    - Authentication
    - Accounts and users

Week 12

- Django
    - Extending the user model
    - URL Router
    - Decorators

    - Forms
    - Deployment

Week 13

- Capstone

Week 14

- Capstone

Week 15

- Capstone

Week 16

- Capstone