Python for Engineers: Part I

Description

In this 4-day training program, students will establish a solid foundation in core Python programming language.

Exercises cover common tasks in:

- data acquisition (via raw input from users, from files, from urls, and from sockets).
- data parsing (including English text, HTML, column-oriented text, CSV and delimited files, JSON, and XML)
- data analysis (using list comprehensions, generator expressions, sets, dictionaries, and named tuples)
- output formatting (using templating to create HTML, vcards, and other output formats).

Themes of the class include:

- learning Python norms, idioms, and programming practices
- developing a clear mental model of the language (magic methods, the object model, memory management, mutability, performance and scaling, etc)
- developing an expressive style that realizes the full advantages of the language
- becoming proficient at rapid application development and repeatable patterns of problem solving

Topics include program organization, principles of object-oriented programming, testing, and debugging. There is special focus on using REST APIs and modern data interchange techniques. The section on socket programming includes HTTP headers and their practical use in REST APIs and networked applications. The examples include a UDP client, UDP server, TCP client, and TCP server. If time allows, the examples are extended with multi-threading and multi-processing.

The style of the class is highly interactive with extensive intensive instructor led problem solving. Students are expected to have high levels of active engagement throughout the class.

Learning Objectives

Although no prior experience with Python is required, this course assumes that students have prior experience with some other programming language such as C, C++, Java, or Perl. This is not an introductory class for absolute beginners on how to program a computer!

Participants should already be familiar with the basic concepts of programming such as variables, data types, statements, control-flow, functions, arrays, data structures, and common programming problems (e.g., searching, sorting, etc.). In addition, it is assumed that students already know how to work with files, folders, editors, command shells, environment settings, internet connections, and other essential aspects of using a computer for software development.

Prerequisites

Although no prior experience with Python is required, this course assumes that students have prior experience with some other programming language such as C, C++, Java, or Perl. This is not an introductory class for absolute beginners on how to program a computer!

Participants should already be familiar with the basic concepts of programming such as variables, data types, statements, control-flow, functions, arrays, data structures, and common programming problems (e.g., searching, sorting, etc.). In addition, it is assumed that students already know how to work with files, folders, editors, command shells, environment settings, internet connections, and other essential aspects of using a computer for software development.

Additional Course Information

Systems Requirements:

This is a hands-on course where students will be expected to write programs. Students are required to bring their own computer (Windows, Macintosh, or Linux) which should have Python 3.7 installed before the first day of class.

The installation instructions can be found here:

https://www.python.org/downloads

Recommended Audience

The audience is broad and includes sales engineers, security specialists, developers, quality assurance engineers, and anyone needing to use a powerful and clean scripting language.

Course Outline

Day 1:

Thorough business level overview of Python (what problems it solves, language philosophy, how it used). Set-up a working interactive environment. Introduce all the core objects and basic mental model of the language. Work through a practical exercise involving CSV parsing, iteration, multiple assignment, the with-statement, try-finally, templating, writing to files, opening urls, and direct use of a REST API.

Day 2:

Cover object-oriented programming. Cover the internal object model including magic methods, memory management, and references. Work on a non-trivial data analysis project. Cover the core objects in greater detail. Parse JSON, XML, and column-oriented text. Demonstrate the effective use of named tuples.

Day 3:

Cover list comprehensions for data analysis. Develop a custom REPL and create a doctest tool from scratch. Cover the core python looping idioms and introduce generators

Day 4:

Cover miscellaneous core language topics such as variable-length argument parsing and argument unpacking. Cover import logic. Build skills parsing data with regular expressions and BeautifulSoup. Learn network programming including details of HTTP headers. Build client and servers using UDP and TCP. If time allows, extend the examples with multi-threading and multi-processing.

Share Course Link:

https://learn.cisco.com?courseId=COT00119819