Python for Networking & Systems Administrators (SysAdmin)

Duration: *4 Day(s)*

Course Overview

Python Essentials for Networking & Systems Administration / SysAdmin is tailored for IT professionals, systems administrators, and network engineers who want to harness the power of Python to simplify and automate everyday tasks across distributed systems. Whether you're new to scripting or looking to expand your skillset, this course provides the perfect opportunity to build essential Python expertise and apply it to real-world scenarios. Working in a hands-on lab environment, you'll start with foundational Python scripting essentials like file operations, regular expressions, and working with binary data, then progress to leveraging network-focused modules such as SSH, Git, and RESTful services. With a strong emphasis on practical application, this course ensures you're not just learning syntax but mastering the tools to solve real challenges in your role.

Unlike quick overviews that leave you scrambling for context, this course emphasizes learning by doing. Through engaging labs and guided exercises, you'll develop tangible skills that translate directly to automating critical tasks like system configuration, network requests, and administrative workflows. Designed for technical professionals who manage distributed systems or oversee network operations, this class equips you to apply Python immediately on the job. By the end of the course, you'll have the confidence and knowledge to use Python as a powerful tool to enhance productivity and efficiency in your day-to-day responsibilities.

Review this course online at https://www.alta3.com/courses/TTPS4824

Objectives

- Automate networking and administrative tasks with Python scripts.
- Work with Python's networking libraries to manage systems and perform diagnostics.
- Handle data efficiently using various Python techniques and formats.
- Develop secure and scalable scripts for real-world applications.

Who Should Attend

- Advanced users
- System administrators
- Website administrators
- Network engineers

Prerequisites

To ensure a smooth learning experience and maximize the benefits of attending this course, you should have the following prerequisite skills:

```
 At least some prior hands-on experience with
scripting or programming. You don't need to be
an expert in either, but you should have had
some exposure and should be coming from a
technical background.
Working with Unix or Linux, and familiarity
with using the command line interface for
simple tasks, such as file navigation and
executing commands.
Basic familiarity working with text editors
like Notepad, or IDEs, would be helpful as the
course includes hands-on lab sessions
requiring code editing.
```

Course Outline

The Python Environment

1. Starting Python
2. If the interpreter is not in your PATH
3. Using the interpreter
4. Trying out a few commands
5. Running Python scripts
6. Getting help
7. Python Editors and IDEs

Variables and Values

8. Using variables
9. Keywords and Builtins
10. Variable typing
11. Strings
12. String operators and methods
13. Numeric literals
14. Math operators and expressions
15. Converting among types

Basic input and output

16. Writing to the screen

17. String Formatting
18. Legacy String Formatting
19. Command line parameters
20. Reading from the keyboard

Flow Control

21. About flow control
22. What's with the white space?
23. if and elif
24. Conditional Expressions
25. Relational Operators
26. Boolean operators
27. while loops
28. Alternate ways to exit a loop

Array types

29. Lists
30. Tuples and unpacking
31. Indexing and slicing
32. Iterating through a sequence
33. Functions for all sequences
34. The range() function
35. List comprehensions
36. Generator Expressions

Working with Files

37. Text file I/O
38. Opening a text file
39. The with block
40. Reading a text file
41. Writing to a text file

Dictionaries and sets

42. When to use dictionaries?
43. Creating dictionaries
44. Getting dictionary values
45. Iterating through a dictionary
46. Reading file data into a dictionary
47. Counting with dictionaries
48. Creating Sets

49. Working with sets

Functions, modules, packages

50. Defining a function
51. Returning values
52. Function parameters
53. Variable scope
54. Creating Modules
55. The import statement
56. Where did **pycache** come from?
57. Module search path
58. Packages

An Introduction to Python Classes

59. About O-O programming
60. Defining classes
61. Constructors
62. Instance methods
63. Properties
64. Class methods and data
65. Static Methods
66. Private methods
67. Inheritance
68. Untangling the nomenclature

Errors and Exception Handling

69. Syntax errors
70. Exceptions
71. Handling exceptions with try
72. Handling multiple exceptions
73. Handling generic exceptions
74. Ignoring exceptions
75. Using else
76. Cleaning up with finally

Efficient Scripting

77. Running external programs
78. Parsing arguments
79. Creating filters to read text files
80. Logging

Regular Expressions

81. Regular Expressions
82. RE Syntax Overview
83. Finding matches
84. RE Objects
85. Compilation Flags
86. Groups
87. Special Groups
88. Replacing text
89. Replacing with a callback
90. Splitting a string

Binary data

91. str vs bytes
92. Binary files
93. Structured binary data
94. Bitwise operations

Network Programming

95. Grabbing a web page
96. Consuming Web services
97. HTTP the easy way
98. sending e-mail
99. Email attachments
100. Remote Access
101. Copying files with Paramiko

Sockets

102. Sockets
103. Socket options
104. Server concepts
105. Client concepts
106. Application protocols
107. Forking servers

Multiprogramming

108. Multiprogramming
109. What Are Threads?
110. The Python Thread Manager

111. The threading Module
112. Threads for the impatient
113. Creating a thread class
114. Variable sharing
115. Using queues
116. Debugging threaded Programs
117. The multiprocessing module
118. Using pools
119. Alternatives to multiprogramming

Serializing Data: XML, XPath, JSON, CSV

120. About XML
121. Normal Approaches to XML
122. Which module to use?
123. Getting Started With ElementTree
124. How ElementTree Works
125. Elements
126. Creating a New XML Document
127. Parsing An XML Document
128. Navigating the XML Document
129. Using XPath
130. About JSON
131. Reading JSON
132. Writing JSON
133. Customizing JSON
134. Reading CSV data
135. Nonstandard CSV
136. Using csv.DictReader
137. Writing CSV Data

Sorting (Bonus Chapters / Time Permitting)

138. Sorting Overview
139. The sorted() function
140. Custom sort keys
141. Lambda functions
142. Sorting nested data
143. Sorting dictionaries
144. Sorting in reverse
145. Sorting lists in place