

APIs and API Design with Python Training

Alta3 Research Authorized Training · Applied Technology Academy

Our OpenStack courses offer both hands-on and lecture, allowing students to learn OpenStack architecture and operation. Students will have 3 attempts to pass our certification exam.

LEVEL	DURATION	EXPERIENCE	AVERAGE SALARY	LABS
Intermediate	5 Days	1 year: Programming	\$121,932	Yes

Course Overview

Application Programming Interfaces (APIs) have become increasingly important as they provide developers with connectivity to everything from rich datasets in an array of formats (such as JSON), to exposing the configurability of software applications and network appliances. Lessons and labs focus on using Python to interact, design, and build APIs for the purposes of scripting automated solutions to complex tasks. Class is mostly live demonstrations and hands on labs.

Develop Python scripts that communicate with RESTful (and non-RESTful) APIs, as well as design RESTful API interfaces themselves. Use Python to open SSH sessions and pass commands to remote servers, move files via SFTP, parse and manipulate popular data structures (JSON, XML, CSV, and YAML), handle errors, interface with the operating system, create highly efficient regular expressions for parsing, and best practice techniques.

Course Outline

- **Lesson 1: Python Review**
 - Lists
 - Dictionaries
 - Tuples
 - Conditionals (if, elif, else)
 - Loops (for and when)
 - Functions
 - Variable Scope – Review
 - Converting boiler plate code to functional code
 - Writing Functions (reusable code)
 - Using pip
 - Useful 3rd party modules
 - Publishing a module
 - Documenting modules
- **Lesson 2: OS Interfacing**

- Operating System interfacing with OS module
- OS module – `listdir()`, `getcwd()`, `makedirs()`
- Common sys Module Attributes & Methods
- Working With the os Module & Files/Directories
- Walking File Trees with `os.walk()`
- File, Path and Directory Examples
- Working with `os.path`
- `os` & `os.path` Module Examples
- Environmental Variables
- Running Shell Commands
- Compressing and archiving (`gzip`, `tar`, `zip`)
- **Lesson 3: Web and RESTful APIs**
 - REST
 - REST APIs and HTTP CRUD
 - REST and OpenStack
 - URI analysis and formation
 - Wireshark capturing
 - cURL
 - Etcd keystore
 - Creating a Python client to interact with API endpoints
 - API dev keys
 - SSH and Python
 - Secure password retrieval
 - Tokens and APIs
- **Lesson 4: Python Protocol Clients**
 - Scripting the browser
 - Scripting with HTTP
 - Creating an HTTP Client & Server
 - Python and SSH
 - Building an SFTP Client & Server
 - Python and SFTP limitations
 - Paramiko for SSH
 - Netmiko and Major Network Vendors (Cisco, Juniper, Arista)
- **Lesson 5: JSON, YAML, XML, CSV and Excel**
 - JSON RFC 7159
 - JSON Formatting
 - YAML intro

- YAML lists
- YAML dictionaries
- YAML line spanning
- Reading YAML is easy
- XML
- CSV
- Import json
- Import yaml
- Decoding json and yaml to use
- Using python to decode data structures like YAML, XML, CSV, and JSON
- Reading from Excel
- Writing to Excel
- **Lesson 6: Generating and Sending Emails**
 - Overview of email modules
 - Creating simple emails
 - Interfacing with your email account
- **Lesson 7: Dates and Times**
 - Python and Cron
 - Import time and time.time()
 - Suspending with sleep()
 - Dealing with time
 - Time formatting
 - Time tuples
 - Creating Calendars
- **Lesson 8: Python Regular Expression (Regex)**
 - Metacharacter review
 - Re modules
 - search() and match()
 - findall()
 - Compiling regex search patterns
 - Creating highly efficient searches
 - Sorting data sets
 - Complex sorts
 - sort() vs sorted()
 - Sorting with functions
 - Applying Regex to file searches
 - Applying Regex to API results

- **Lesson 9 : Code Review**
 - Best practice
 - Using pylint
 - Conventions
 - Underscore
 - Double underscore
 - Monkey Patching
- **Lesson 10: Web API Design with Flask**
 - Flask Overview
 - Decorators
 - Building APIs with Python and Flask
 - APIs returning Jinja2 templating
 - Returning a 'cookie'
 - Building Sessions
 - Redirecting from URIs
 - Build an API to accept a file upload
- **Lesson 11: SQLite**
 - Overview
 - Connecting to Python
 - Read / Write operations
 - Other useful instructions
 - Connecting APIs and SQLite
 - Reading and Writing to Databases with APIs
- **Lesson 12: Processes and Threads**
 - Threading
 - Context change
 - Deadlock errors
 - Thread starvation
 - Racing conditions and racing specifics
 - Working with Locks
 - Lab 1 – Using vim
 - Lab 2 – Making a Github account
 - Lab 3 – Getting dir(obj) help() and pydoc
 - Lab 4 – Lists
 - Lab 5 – Dictionaries
 - Lab 6 – Python Data to JSON file
 - Lab 7 – Python Data to YAML file

- Lab 8 – import time
- Lab 9 – List and Dict Modeling
- Lab 10 – try and except
- Lab 11 – Construct a SimpleHTTPServer and HTTP Client
- Lab 12 – RESTful Open APIs
- Lab 13 – APIs and JSON Decode
- Lab 14 – requests library
- Lab 15 – APIs and Dev Keys
- Lab 16 – RESTful APIs and Dev Keys
- Lab 17 – getpass – Secure Password Retrieval
- Lab 18 – Paramiko SFTP with UN and PW
- Lab 19 – Paramiko SSH with RSA
- Lab 20 – Scripting Commands over SSH
- Lab 21 – Argument Parsing
- Lab 22 – Making Excel Sheets
- Lab 23 – Reading Excel Sheets
- Lab 24 – Timestamping Data
- Lab 25 – API Tokens and Subjects
- Lab 26 – RegEx to Check IP Address
- Lab 27 – Use RegEx to Search Text
- Lab 28 – Search and Replace Data
- Lab 29 – Compiling Search Objects
- Lab 30 – Testing of a Match Exists
- Lab 31 – Getting sorted()
- Lab 32 – Sort Stability and Complex Sorts
- Lab 33 – CSV data
- Lab 34 – Unpacking Arguments
- Lab 35 – XML Parsing with ElementTree
- Lab 36 – Automating SMTP and Extended SMTP (Email)
- Lab 37 – Archive with zipfile
- Lab 38 – Building APIs with Python
- Lab 39 – Flask APIs and Jinja2
- Lab 40 – Flask APIs and Cookies
- Lab 41 – Flask Sessions
- Lab 42 – Flask Redirection and Errors
- Lab 43 – Flask Uploading Files
- Lab 44 – Connecting an API to a Database
- Lab 45 – Learning sqlite3

- Lab 46 – Tracking API Data with sqlite3
- Lab 47 – Tracking Inventory with sqlite
- Lab 48 – Working with Threads

Intended Audience

System administrators, network engineers, and developers will find this course compelling as they build and interact APIs that not only return highly parsable datasets, but also trigger scripted actions. Some previous experience with Python is ideal, although, coding experience in another language is also enough to find success within this course.

Prerequisites

- Recommended Prerequisite: Python Basics (5 days)
- Coding experience in another language serves as an adequate prerequisite

Follow-On Courses

Python for Network Automation

Ready to enroll? Call 800.674.3550 or email Info@AppliedTechAc.com — private team cohorts, GSA MAS purchasing, military credentialing funding, VA GI Bill and VR&E and student financing available.