Training 4 Programmers

To discuss this course and customizations:
Call: +1 434-509-5680 or Email: sales@training4programmers.com

Task Automation with Python

Duration

2 days

Description

This Task Automation with Python training course is designed to equip computer programming professionals with the skills and knowledge required to automate tasks using Python. The course covers a wide range of topics, from setting up a development environment and understanding the importance of logging to mastering file and folder operations and exploring concurrent and network programming. Participants will also learn about Generative AI programmer tooling, environment variables, command line arguments, and regular expressions. The course includes optional customization to suit individual or organizational needs, making it a flexible learning solution for professionals looking to enhance their Python programming skills. Possible customizations include integrating OpenAI and generative AI, working with tabluar data and libraries like Pandas and Polars, and much more!

Objectives

- Understand the concept of task automation and the benefits of using Python for this purpose.
- Set up a development environment for Python script programming, including configuration of Visual Studio Code and Python code linting.
- Learn about Generative AI tooling and how to use GitHub Copilot for Python programming and terminal commands.
- Understand the importance of logging in Python, including the use of the Python logging module, log levels, handlers, and formatters.
- Learn about exit codes in Python, how to return, capture, and handle them.
- Understand the role of environment variables in Python and how to set and read them.
- Gain proficiency in file and folder operations in Python, including reading and writing files, working with various file formats, and high-level file operations.
- Learn about concurrent and network programming in Python, including orchestrating subprocesses, accessing data sources asynchronously, and integrating with various API server integrations.
- Explore the basics of Python Celery (RabbitMQ) or Python Faust (Kafka) for distributed task automation.

Prerequisites

All students must be able to comfortably write Python scripts using basic data types, program structures, and the standard Python library.

Training Materials

All students receive comprehensive courseware covering all topics in the course. The instructor distributes courseware via GitHub. The courseware includes documentation and extensive code samples. Students practice the topics covered through challenging hands-on lab exercises.

Software Requirements

Students will need a free, personal GitHub account to access the courseware. All students will need a modern web browser such as Google Chrome. Student machines will need a text editor, such as VS Code, the latest LTS version of Node.js, the latest Python version, PanDoc, and OpenOffice. Students will need permission to install NPM packages and PyPi packages. Preconfigured student virtual machines can provided upon request. For some demostrations, students will need access to 3rd party services such as AWS, Azure, and GitHub. The need for these services will be discussed as part of planning the specifics of the class.

Outline

- Introduction
  - What is Task Automation?
  - Why Use Python for Task Automation?
  - Python Automation Libraries
  - Python Automation Tools
  - Python Automation Best Practices
  - Task Automation Examples and Ideas
- Development Environment
  - Configure Visual Studio Code for Python Script Programming
  - Python Code Linting & Reformatting with Ruff & MyPy
  - Debugging Python Scripts with Visual Studio Code
  - Docker Desktop
- Generative AI Tooling (optional)
  - GitHub Copilot and Visual Studio Code Integration
  - Using GitHub Copilot for Python Programming
  - Using GitHub Copilot for Terminal Commands
  - Leverage ChatGPT to Program Task Automation Scripts
  - **Customize the Course with Your Generative AI Programmer Tool (optional)**

- Logging
  - What is Logging?
  - Benefits of Logging
  - Python Logging Module
  - Log Levels
  - Log Handlers
  - Log Formatters
  - **Customize the Course with Your Logging Destination (optional)**
- Exit Codes
  - What are Exit Codes?
  - Return Exit Codes
  - Capture Exit Codes
  - Handle Exit Codes
  - Error Handling, Exit Codes, and Logging
- Environment Variables
  - What are Environment Variables?
  - Setting Environment Variables for a Script (Windows, macOS, and Linux)
  - Reading Environment Variables
  - Setting Environment Variables in Python for Child Processes
- Files, Folders, and Operating System
  - OS Module vs PathLib Module
  - Current Working Directory
  - File / Folder Paths
  - Reading and Writing Files
  - Asynchronous Reading and Writing Files
- Document Processing
  - Work with Various File Formats such as CSV, JSON, and YAML
  - Read and Write Excel Files
  - Convert from Markdown to Word
  - Convert from Word to PDF
  - Modify Word Files
  - Modify PDF Files
  - **Customize the Course with Your Preferred File Formats (optional)**
- High-Level File Operations
  - ShUtil Module
  - Copy/Move/Delete Files

  - Copy/Move/Delete Directory Tree
  - Determine Disk Usage
  - Working with File Archives
  - Create, Remove, & List Directories
- Command Line Arguments
  - Arg Parse Module
  - Click Package
  - Positional Arguments
  - Named Arguments
  - Default Arguments
  - Argument Data Types
  - Argument Documentation
- Regular Expressions
  - RE syntax overview
  - RE objects
  - Searching and matching
  - Compilation flags
  - Groups and special groups
  - Replacing text
  - Splitting strings
- Subprocess
  - Run Method
  - POpen Class
  - Running Subprocesses
  - Capture and Use Subprocess Output
  - Exit Codes
- Concurrent and Network Programming
  - Orchestrating Subprocesses
  - Access Data Sources Asynchronously
  - Send Emails
  - Use SFTP to Download/Upload Files
  - Download/Upload Files from/to S3 Buckets (optional)
  - Download/Upload Files from/to Azure Blob Storage (optional)
  - Calling OpenAI API with Python (optional)
  - AI Embeddings, RAG, and PostgreSQL with Vector Extension (optional)
  - Integration with GitHub Rest API and Web Hooks (optional)

  - Task Automation in GitHub Actions via Docker (optional)
  - **Customize the Course with Your Preferred API Server Integrations (optional)**
- Quick Exploration of Distributed Task Automation (choose Faust/Kafka or Celery/RabbitMQ)
- Explore Stream Processing with Python Faust (optional)
  - Introduction to Stream Processing
  - What is Faust?
  - Faust Architecture
  - Apache Kafka
  - Faust Configuration
  - Agents and Streams
  - Table and Windowing
  - Asynchronous Processing
  - Deployment and Scaling
- Explore Distributed Task Automation with Celery (optional)
  - What is Distributed Task Automation?
  - What is Celery?
  - Celery Architecture
  - RabbitMQ and PostgreSQL
  - Celery Configuration
  - Tasks and Queues
  - Orchestrating Tasks
  - Asynchronous Tasks
  - Task Worker Containers
- Conclusion
  - Summary of Key Concepts
  - Q&A
  - Further Resources and Next Steps