



To discuss this course and customizations:  
Call: 434-509-5680 or Email: [sales@cloudcontraptions.com](mailto:sales@cloudcontraptions.com)

## Distributed Task Automation with Python Faust and Kafka

### Duration

16 hours

### Target Audience

- Proficiency in Python programming, including experience with Python 3.x.
- Familiarity with basic concepts of distributed systems and task automation.
- Experience with Docker and containerization concepts is beneficial but not required.
- Basic understanding of message brokers and stream processing concepts is helpful but not required.
- All students should have taken the Python Task Automation course or have significant experience with the topics covered in the Python Task Automation course.

### Executive Summary

This course equips Python developers with practical expertise in distributed task automation using Python Faust and Kafka. Learn to build scalable, real-time data pipelines with Kafka for messaging and Faust for stream processing. Set up Docker-based environments, manage Kafka clusters, and deploy fault-tolerant, stateful applications. Ideal for professionals ready to scale automation and streamline backend architecture with modern distributed systems.

### Description

This comprehensive course guides experienced Python developers through every aspect of building resilient, high-performance distributed task pipelines with Python Faust and Apache Kafka. You'll start by exploring the fundamentals of task automation before diving into hands-on environment setup—installing Python tools, containerizing your applications with Docker, and standing up a Kafka cluster. From there, you'll master Faust's powerful real-time stream processing API, learning to manage state, ensure fault



To discuss this course and customizations:  
Call: 434-509-5680 or Email: [sales@cloudcontraptions.com](mailto:sales@cloudcontraptions.com)

tolerance, and handle errors gracefully. We'll then show you how to monitor your applications, tune performance, and scale seamlessly in production, with best practices for deployment and observability. Finally, you'll put it all together in a capstone project: designing and implementing a fully functional, real-time data pipeline using Faust and Kafka. Along the way, we'll tailor examples to your domain so you leave with immediately applicable skills for automating complex workflows at scale.

## Objectives

- Understand the concept and application of Distributed Task Automation.
- Set up and configure a Python development environment for script programming.
- Learn the basics of containerization and how to use Docker for creating and running containers.
- Gain in-depth knowledge of Kafka, its architecture, and how to set up a Kafka cluster.
- Master the basics and advanced concepts of Python Faust, including agents, stream processing, state management, and fault tolerance.
- Learn how to monitor and manage Kafka and Faust applications, including error handling and retry logic.
- Understand the best practices for deploying Kafka and Faust in production, ensuring high availability and optimizing performance.
- Implement a real-time data pipeline with Faust and Kafka.

## Duration

16 hours of intensive training with live instruction delivered over two to four days to accommodate varied scheduling needs.

## 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. 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 like Visual Studio Code, the latest Python version, Docker Desktop, PanDoc, and OpenOffice. Students will need permission to



To discuss this course and customizations:  
Call: 434-509-5680 or Email: [sales@cloudcontraptions.com](mailto:sales@cloudcontraptions.com)

install NPM and PyPi packages as well as the ability to download Docker images. Preconfigured student virtual machines can be provided upon request.

## Software Requirements

Students will need a free, personal GitHub account to access the courseware. Student machines will need a text editor like Visual Studio Code, the latest Python version, Docker Desktop, Pandoc, and OpenOffice. Students will need permission to install NPM and PyPi packages as well as the ability to download Docker images. Preconfigured student virtual machines can be provided upon request.

## Training Topics

### Overview of Distributed Task Automation

- What is Distributed Task Automation?
- Overview of Python Faust
- Faust compared to Celery
- What is Streaming?
- What is Kafka?
- What is Zookeeper?
- Kafka + Zookeeper compared to RabbitMQ + PostgreSQL

### 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

### Containerization

- What is a Container?
- What is Docker?
- What is Docker Hub?
- Images and Containers
- Create an Image with Dockerfile
- Run Containers
- Configure Containers with Environment Variables
- Docker Compose
- Docker Compose Networking



To discuss this course and customizations:  
Call: 434-509-5680 or Email: [sales@cloudcontraptions.com](mailto:sales@cloudcontraptions.com)

- Docker Compose Volume

### Scaling Faust Applications

- Parallelism and Partitioning in Kafka
- Running Multiple Faust Workers

### Monitoring and Management

- Monitoring Kafka and Faust
- Using Kafka Monitoring Tools (e.g., Kafka Manager, Confluent Control Center)
- Logging and Metrics in Faust
- Handling Errors and Retries
- Configuring Error Handling in Faust
- Implementing Retry Logic

### Scaling and Deployment

- Deploying Kafka and Faust in Production
- Best Practices for Kafka Cluster Deployment
- Deploying Faust Apps with Docker and Kubernetes
- High Availability and Fault Tolerance
- Configuring Kafka for High Availability
- Ensuring High Availability in Faust Applications

### Performance Tuning

- Kafka Performance Tuning
- Optimizing Faust Performance
- Implementing a Real-Time Data Pipeline with Faust and Kafka

### Conclusion

- Summary of Key Concepts
- Q&A
- Further Resources and Next Steps