Document Generated: 09/19/2024

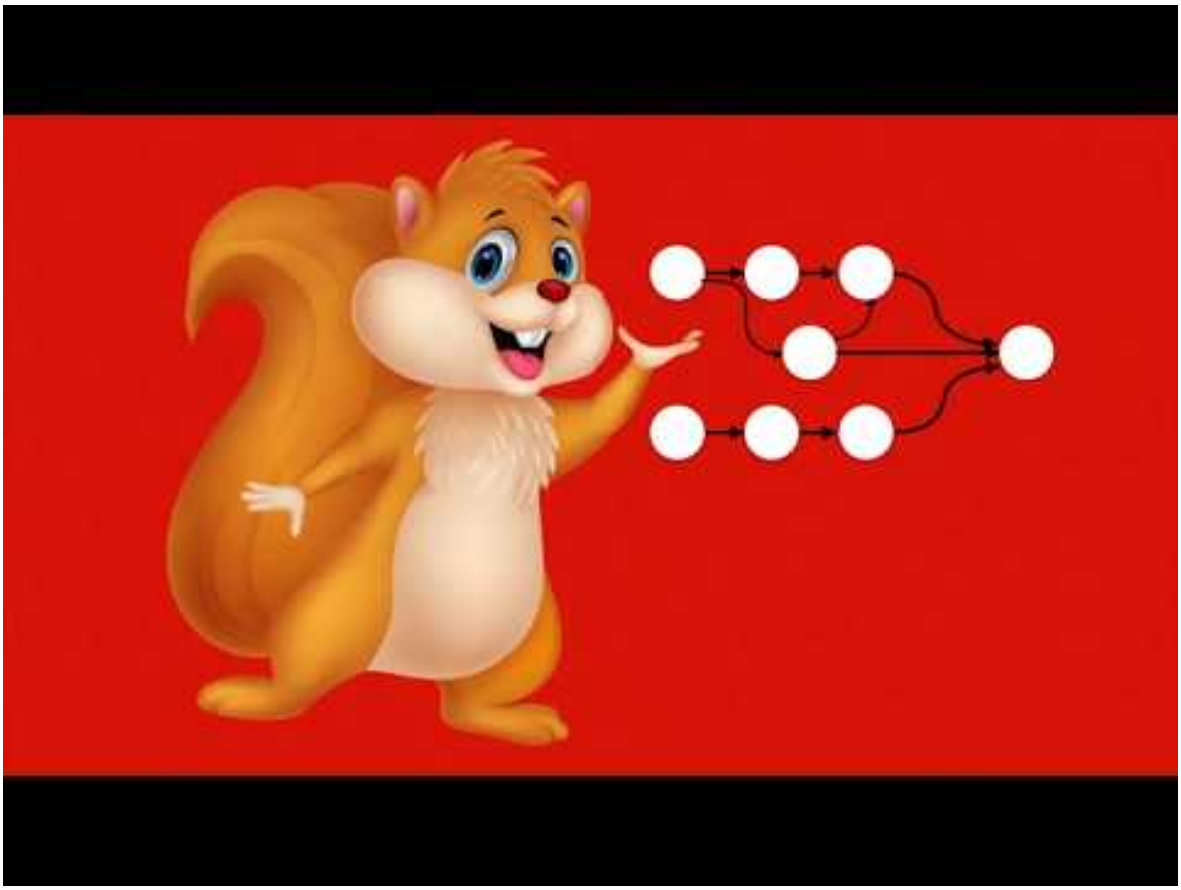
Learning Style: On Demand

Provider: Java

Difficulty: Intermediate

Course Duration: 3 Hours

Learn By Example: Apache Flink



About This Course:

Flink is a stream processing system with an added ability to do many other things, like machine learning, graph algorithms, batch processing, etc. Using Flink you can create apps that allow you to be extremely sensitive to the latest data, such as tracking spikes in payment gateway failure or triggering live stock price movements.

This program has Thirty Solved Examples for developing both batch processing and streaming apps

Course Objective:

· Multiple-stream operations: cogroup, union, comap, connect, iterate and join

· DataStream API Transformations: Map, Filter, Reduce and FlatMap

· Window operations: Tumbling, Sliding, Session and Count windows; time notion and how custom Window functions are implemented

· DataSet API transformations: reduce, map, filter, reduce the Group

· Managing System and Checkpointing fault tolerance

· Using Gelly to represent Graph data

· Usage of Flink-ML to apply ML algorithms on the fly

Audience:

· People who are familiar with Batch processing techniques such as Hadoop want to know more about Stream processing

· Engineers who are currently looking to configure end-to-end data processing pipelines that adapt to real-time changes

Prerequisite:

· You can install a Java IDE, such as IntelliJ Idea

· Java programming expertise and familiarity with the use of Java frameworks

· Creating Jars with Maven, debugging and compiling Java code

Course Outline:

Introduction

You, This Course and Us

Flink's Stream Processing Architecture

What is stream processing?
Stream processing vs Batch processing
Requirements of a Streaming Architecture
Stream processing with Apache Flink
Resources

Getting Started with Flink

Installing Flink
Setting up your Flink project with Maven

Hello World!

Data Representation and Programming Model
Example 1: Writing a Flink program

Transformations using the DataStream API

Example 2: The Filter operation
Example 3: The Map operation
Example 4: The FlatMap operation
Stateless and Stateful Transformations
Keyed Streams
Example 5: Creating a stream of Tuples
Example 6: Transformations on Keyed Streams
Example 7: Number aggregations
Example 8: The Reduce Operation

Window Operations

Windows Transformation
Example 9 and 10: Keyed vs NonKeyed, Sliding and Tumbling Windows
Example 11: Count Windows
Example 12: Session Windows
Understanding the Window API
Example 13: Implementing a Custom Window Function
Example 14: Changing the time characteristic
Example 15: Twitter Streaming Window

Custom Sources

Example 16: Custom Sources

State and Checkpointing

Example 17: Value State
Example 18: List State
Example 19: Reducing State
Example 20: Checkpointing and Restart strategies

Operations on Multiple Streams

Example 21: Unions
Example 22: Joining Streams
Example 23: coGroup
Example 24: coMap
Example 25: Iterate
Example 26: Split

Transformations in the DataSet API

Example 27: Applying Transformations on DataSets

Graph representation with Gelly

Example 28: Representing Graph data

Machine Learning with Flink ML

Example 29: Scala Hello World

Credly Badge:

Display your Completion Badge And Get The Recognition You Deserve.

Add a completion and readiness badge to your Linkedin profile, Facebook page, or Twitter account to validate your professional and technical expertise. With badges issued and validated by Credly, you can:

- Let anyone verify your completion and achievement by clicking on the badge
- Display your hard work and validate your expertise
- Display each badge's details about specific skills you developed.

Badges are issued by QuickStart and verified through Credly.

Find Out More or See List Of Badges