The MathServices App Case Study: Overview

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Part of the Lesson

- Understand how various Java concurrency frameworks are applied in a case study using Spring WebMVC to perform a pair of math services

See github.com/douglascraigschmidt/LiveLessons/tree/master/WebMVC/ex3
Overview of the Math Services App Case Study
Overview of the MathServices App Case Study

- This case study shows how to use Spring WebMVC to send & receive HTTP GET requests synchronously to/from parallel clients & multiple microservices.

Three Java concurrency models are applied in this case study.
Overview of the MathServices App Case Study

- This case study shows how to use Spring WebMVC to send & receive HTTP GET requests synchronously to/from parallel clients & multiple microservices

MathServicesDriver

The client sends requests in parallel using Java structured concurrency (StructuredTaskScope)

See github.com/douglascraigschmidt/LiveLessons/tree/master/WebMVC/ex3/client
Overview of the MathServices App Case Study

• This case study shows how to use Spring WebMVC to send & receive HTTP GET requests synchronously to/from parallel clients & multiple microservices

Two microservices receive requests in bulk & process them in parallel using Java structured concurrency (Thread PerTaskExecutor) & parallel streams

See github.com/douglascraigschmidt/LiveLessons/tree/master/WebMVC/ex3/server
Structure of the MathServices App Project
Structure of the MathServices App Project

• The MathServices App project source code is organized into several modules & packages

See github.com/douglascraigschmidt/LiveLessons/tree/master/WebMVC/ex3
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
      - Contains the “app” entry points, the controllers, & the services implementation strategies
    - Showcases both Java structured concurrency (ThreadPerTaskExecutor) & Java parallel streams
The MathServices App project source code is organized into several modules & packages:

- **main**
  - microservices
  - common
  - Consolidates various project-specific helper classes
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
    - common
  - utils
    - Consolidates various reusable helper classes
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
    - common
    - utils
  - resources
    - Defines various application properties
      - e.g., microservice names & port numbers
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
      - This test driver causes the client to send/receive requests/responses to/from the microservices running on the server & displays the results
    - Showcases Java structured concurrency (StructuredTaskScope)
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
  - Sends HTTP GET requests to the server using various Java frameworks
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
  - client
  - utils
    - Consolidates various reusable helper classes
Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
  - client
  - utils
    - resources
      - Defines various application properties
        - e.g., disable/enable logging
Pros & Cons of the MathServices App
Pros & Cons of the MathServices App

• Pros
• Each microservice runs in its own process (& potentially its own computer in a data center or cloud environment)

Can improve system scalability & reliability
Pros & Cons of the MathServices App

- **Cons**
  - Client(s) must be explicitly programmed to connect & communicate with each microservice explicitly.
  - Complicates configuration, deployment, testing, & security.
End of the MathServices App Case Study: Overview