The MathServices App Case Study: Overview

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

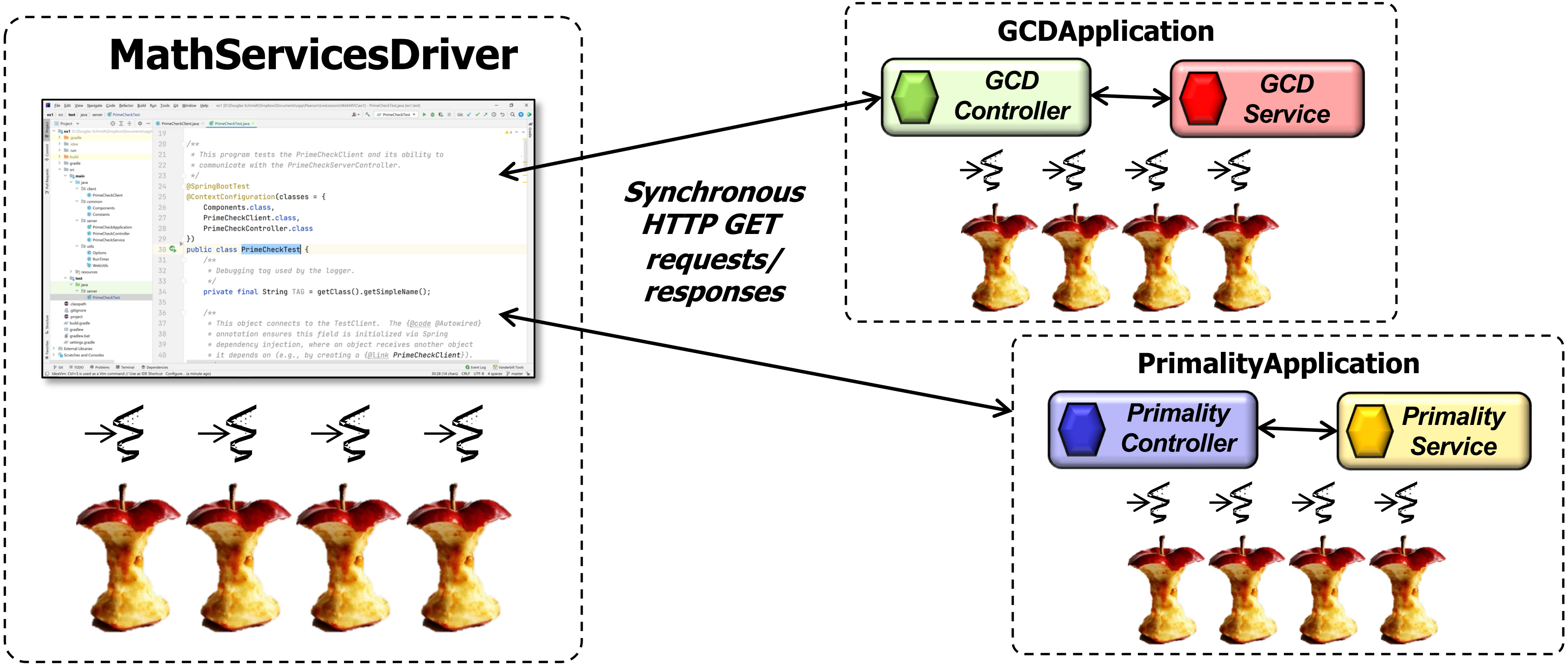
Professor of Computer Science

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



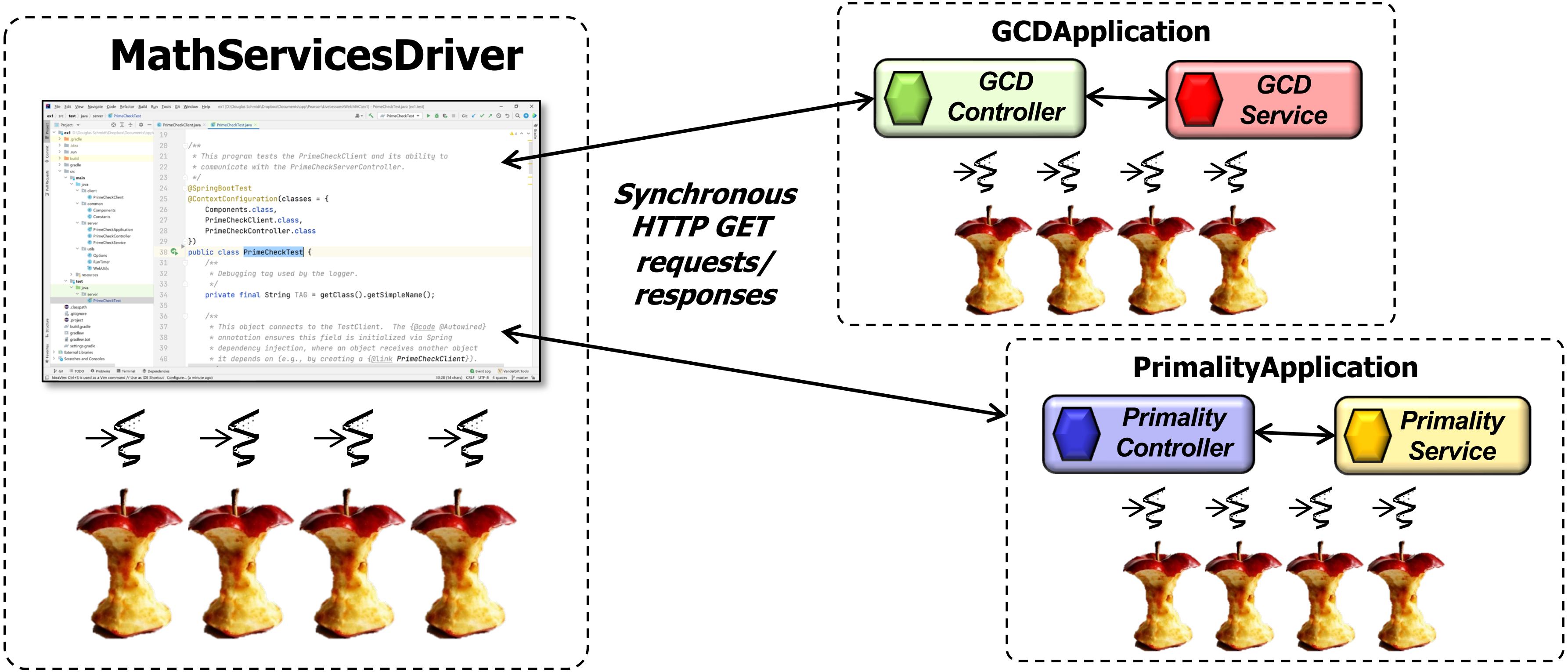
```java

/**
 * This program tests the PrimeCheckClient and its ability to
 * communicate with the PrimeCheckServerController.
 */
@SpringBootTest
@ContextConfiguration(classes = {
    Components.class,
    PrimeCheckClient.class,
    PrimeCheckController.class
})
public class PrimeCheckTest {
    /**
     * Debugging tag used by the logger.
     */
    private final String TAG = getClass().getSimpleName();

    /**
     * This object connects to the TestClient.  The {@code @Autowired}
     * annotation ensures this field is initialized via Spring
     * dependency injection, where an object receives another object
     * it depends on (e.g., by creating a {@link PrimeCheckClient}).
```

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



```java
/**
 * This program tests the PrimeCheckClient and its ability to
 * communicate with the PrimeCheckServerController.
 */
@SpringBootTest
@ContextConfiguration(classes = {
    Components.class,
    PrimeCheckClient.class,
    PrimeCheckController.class
})
public class PrimeCheckTest {
    /**
     * Debugging tag used by the logger.
     */
    private final String TAG = getClass().getSimpleName();

    /**
     * This object connects to the TestClient.  The {@code @Autowired}
     * annotation ensures this field is initialized via Spring
     * dependency injection, where an object receives another object
     * it depends on (e.g., by creating a {@link PrimeCheckClient}).
```

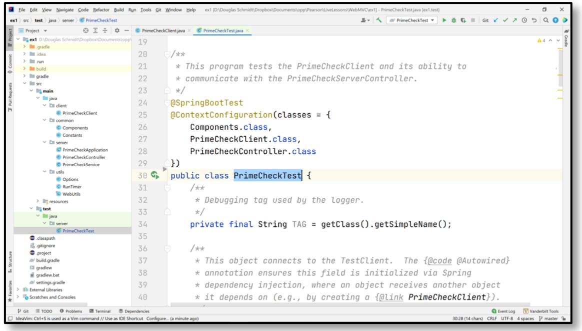
Synchronous HTTP GET requests/ responses

Three Java concurrency models are applied in this case study

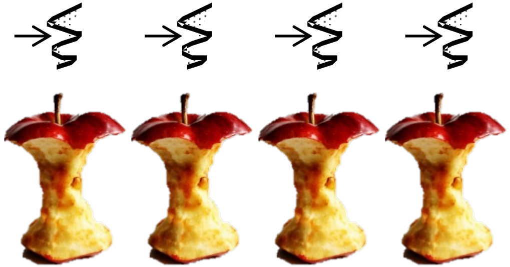
Overview of the MathServices App Case Study

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

MathServicesDriver



```
/**
 * This program tests the PrimeCheckClient and its ability to
 * communicate with the PrimeCheckServerController.
 */
@SpringBootTest
@ContextConfiguration(classes = {
    Components.class,
    PrimeCheckClient.class,
    PrimeCheckController.class
})
public class PrimeCheckTest {
    /**
     * Debugging tag used by the logger.
     */
    private final String TAG = getClass().getSimpleName();

    /**
     * This object connects to the TestClient.  The {@code @Autowired}
     * annotation ensures this field is initialized via Spring
     * dependency injection, where an object receives another object
     * it depends on (e.g., by creating a {@link PrimeCheckClient}).
```

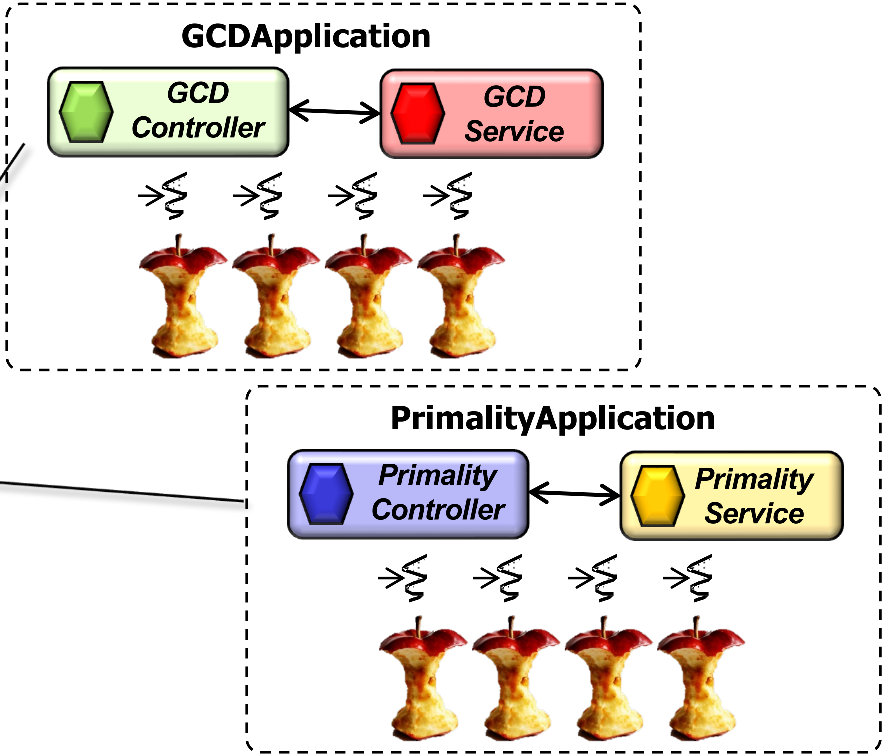


The client sends requests in parallel using Java structured concurrency (StructuredTaskScope)

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



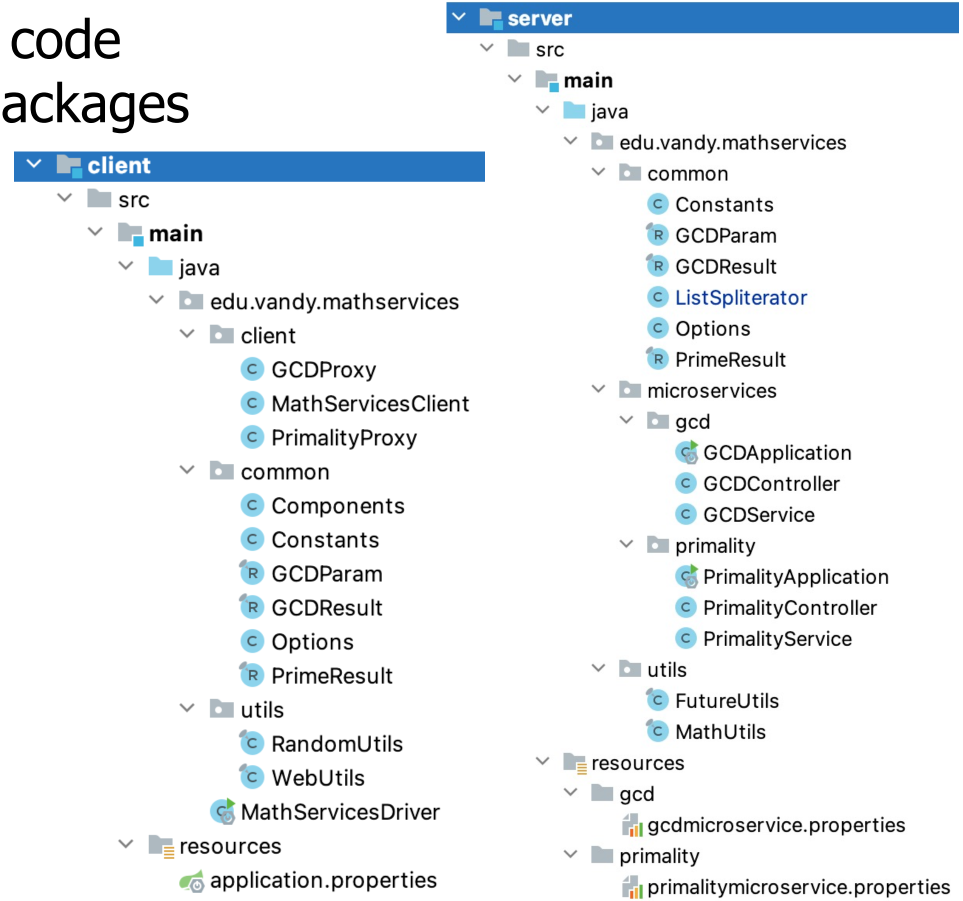
Two microservices receive requests in bulk & process them in parallel using Java structured concurrency (ThreadPerTaskExecutor) & parallel streams

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



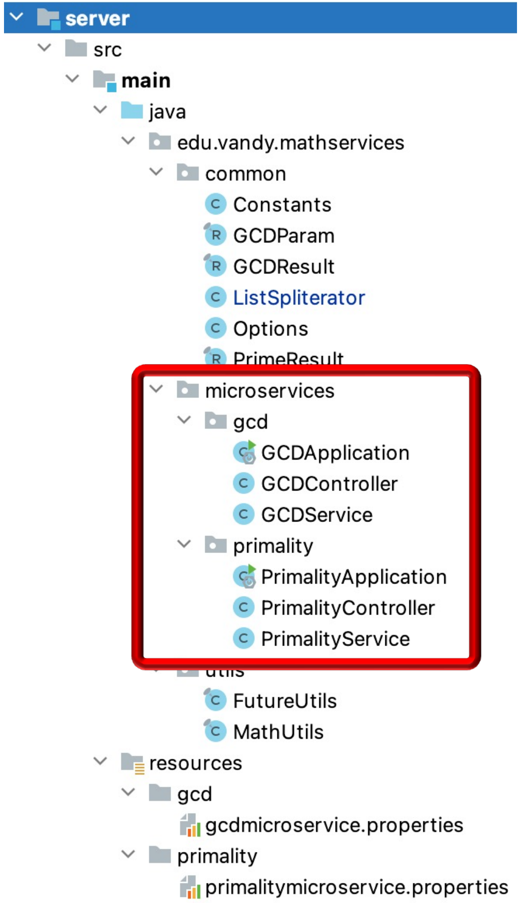
```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

```
server
  src
    main
      java
        edu.vandy.mathservices
          common
            Constants
            GCDParam
            GCDResult
            ListSpliterator
            Options
            PrimeResult
          microservices
            gcd
              GCDApplication
              GCDController
              GCDService
            primality
              PrimalityApplication
              PrimalityController
              PrimalityService
          utils
            FutureUtils
            MathUtils
      resources
        gcd
          gcdmicroservice.properties
        primality
          primalitymicroservice.properties
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/WebMVC/ex3

Structure of the MathServices App Project

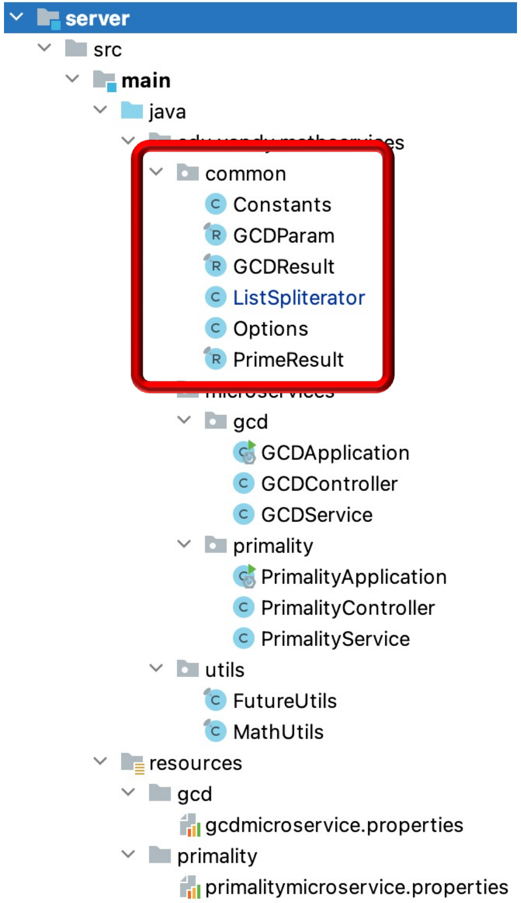
- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
      - Contains the "app" entry points, the controllers, & the services implementation strategies
        - Showcases both Java structured concurrency (ThreadPerTaskExecutor) & Java parallel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



```
server
  src
    main
      java
        edu.vandy.mathservices
          common
            Constants
            GCDParam
            GCDResult
            ListSpliterator
            Options
            PrimeResult
          microservices
            gcd
              GCDApplication
              GCDController
              GCDService
            primality
              PrimalityApplication
              PrimalityController
              PrimalityService
          utils
            FutureUtils
            MathUtils
      resources
        gcd
          gcdmicroservice.properties
        primality
          primalitymicroservice.properties
```

Structure of the MathServices App Project

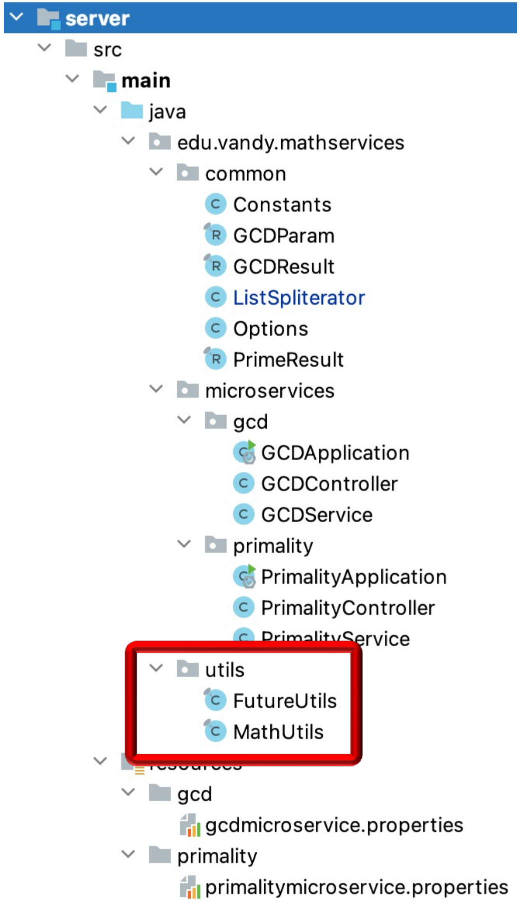
- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
    - common
      - Consolidates various project-specific helper classes



```
server
  src
    main
      java
        edu.vandy.mathservices
          common
            Constants
            GCDParam
            GCDResult
            ListSpliterator
            Options
            PrimeResult
          microservices
            gcd
              GCDApplication
              GCDController
              GCDService
            primality
              PrimalityApplication
              PrimalityController
              PrimalityService
          utils
            FutureUtils
            MathUtils
      resources
        gcd
          gcdmicroservice.properties
        primality
          primalitymicroservice.properties
```

Structure of the MathServices App Project

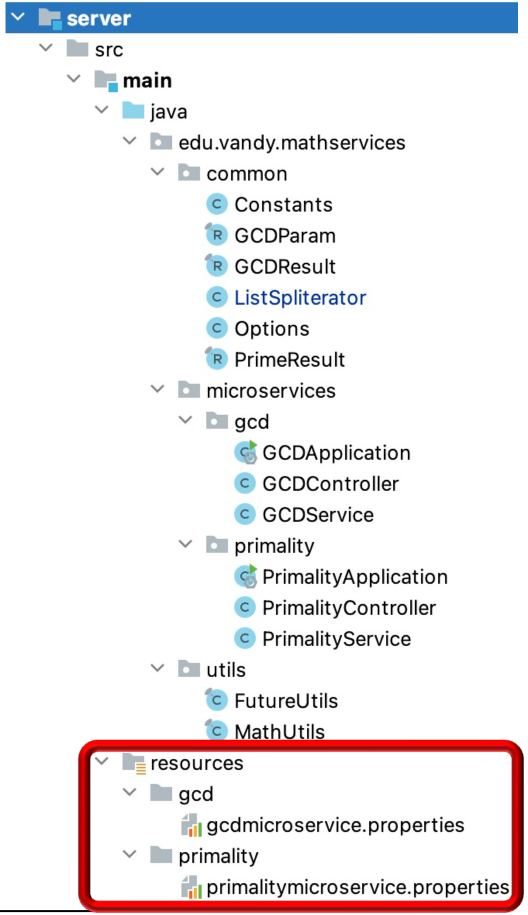
- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
    - common
    - utils
      - Consolidates various reusable helper classes



```
server
  src
    main
      java
        edu.vandy.mathservices
          common
            Constants
            GCDParam
            GCDResult
            ListSpliterator
            Options
            PrimeResult
          microservices
            gcd
              GCDApplication
              GCDController
              GCDService
            primality
              PrimalityApplication
              PrimalityController
              PrimalityService
          utils
            FutureUtils
            MathUtils
      resources
        gcd
          gcdmicroservice.properties
        primality
          primalitymicroservice.properties
```

Structure of the MathServices App Project

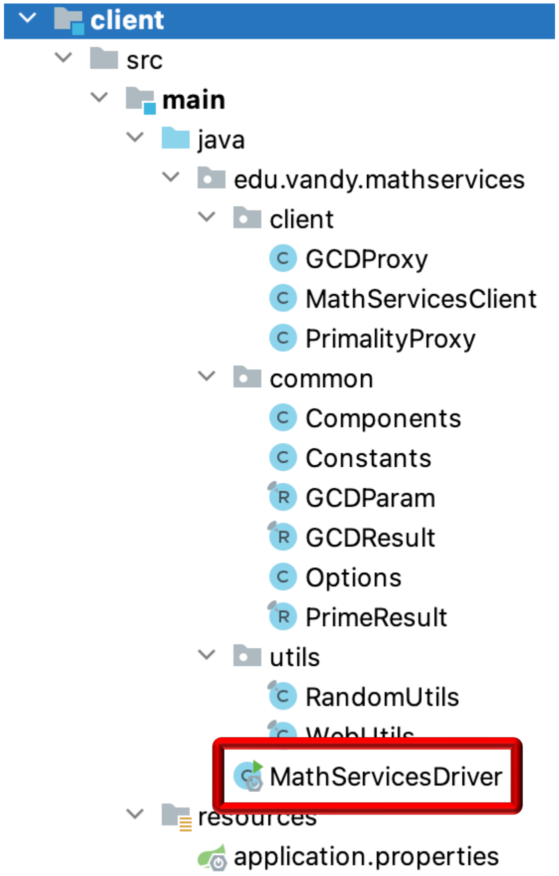
- The MathServices App project source code is organized into several modules & packages
  - main
    - microservices
    - common
    - utils
    - resources
      - Defines various application properties
        - e.g., microservice names & port numbers



```
server
  src
    main
      java
        edu.vandy.mathservices
          common
            Constants
            GCDParam
            GCDResult
            ListSpliterator
            Options
            PrimeResult
          microservices
            gcd
              GCDApplication
              GCDController
              GCDService
            primality
              PrimalityApplication
              PrimalityController
              PrimalityService
          utils
            FutureUtils
            MathUtils
      resources
        gcd
          gcdmicroservice.properties
        primality
          primalitymicroservice.properties
```

Structure of the MathServices App Project

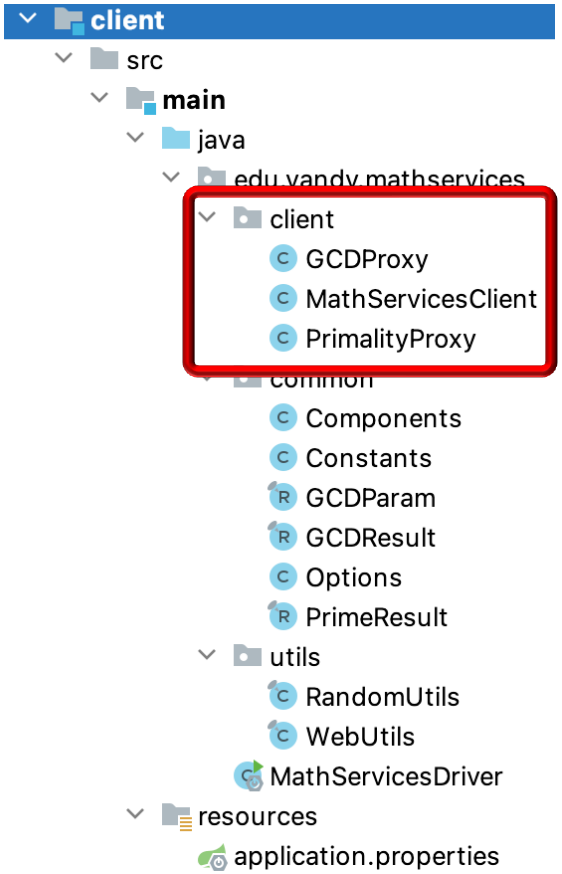
- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
      - This test driver causes the client to send/receive requests/responses to/from the microservices running on the server & displays the results
        - Showcases Java structured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



```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

Structure of the MathServices App Project

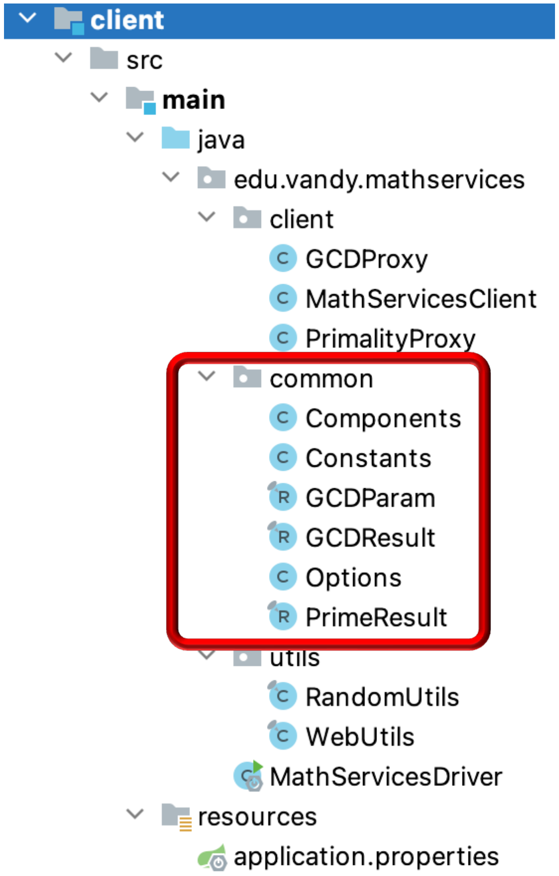
- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
    - client
      - Sends HTTP GET requests to the server using various Java frameworks



```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

Structure of the MathServices App Project

- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
    - client
    - common
      - Consolidates various project-specific helper classes

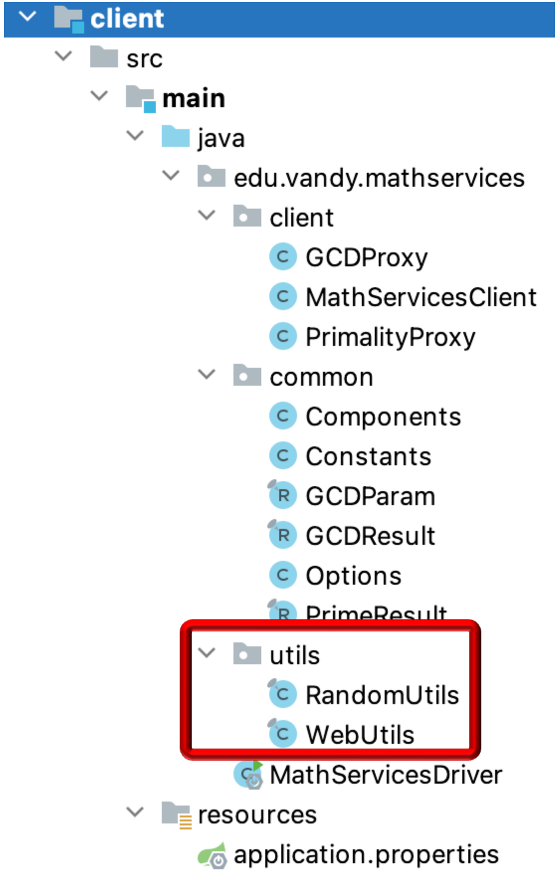


```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

These helper classes should be factored into a separate module

Structure of the MathServices App Project

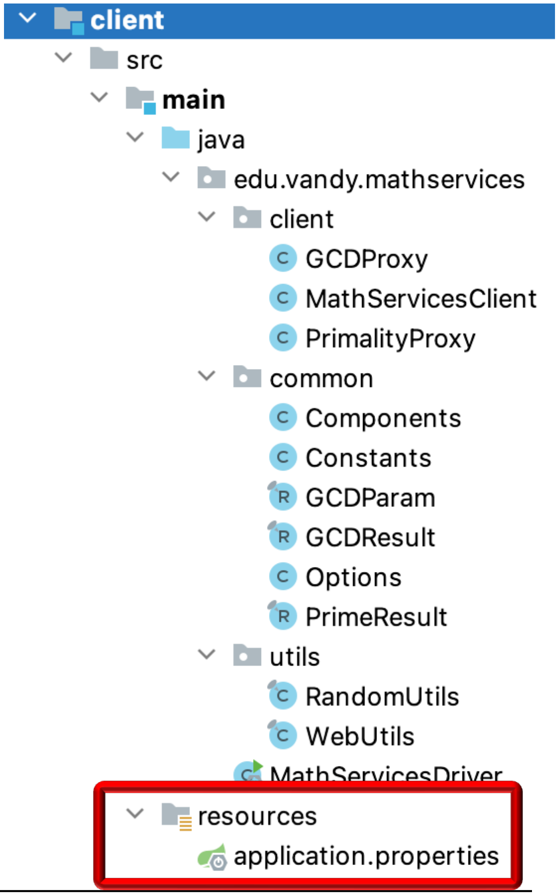
- The MathServices App project source code is organized into several modules & packages
  - client
    - MathServicesDriver
    - client
    - common
    - utils
      - Consolidates various reusable helper classes



```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

Structure of the MathServices App Project

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

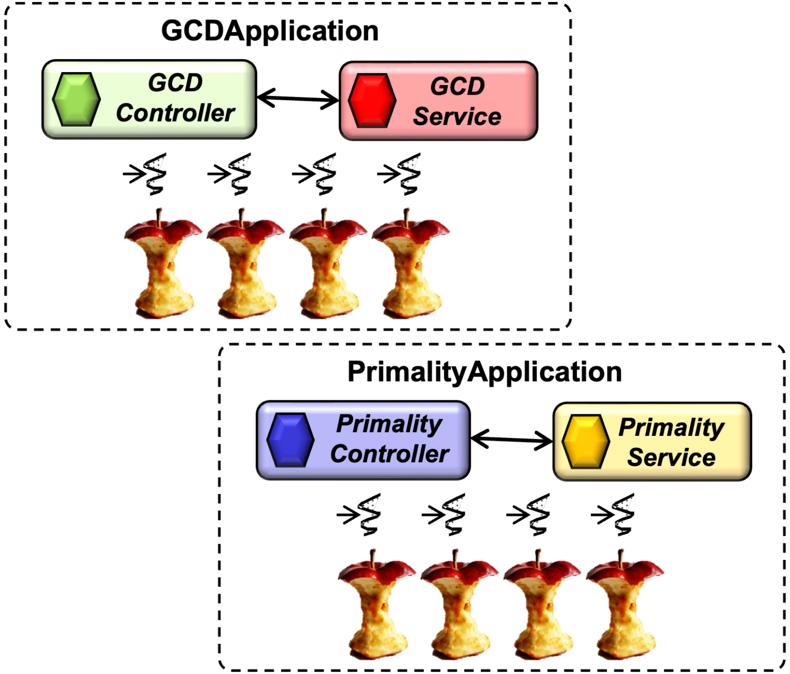


```
client
  src
    main
      java
        edu.vandy.mathservices
          client
            GCDProxy
            MathServicesClient
            PrimalityProxy
          common
            Components
            Constants
            GCDParam
            GCDResult
            Options
            PrimeResult
          utils
            RandomUtils
            WebUtils
          MathServicesDriver
      resources
        application.properties
```

Pros & Cons of the MathServices App

Pros & Cons of the MathServices App

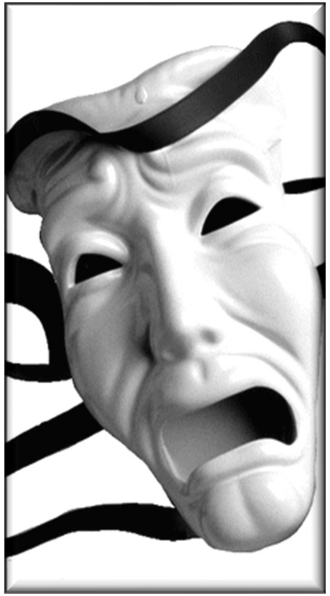
- Pros
  - Each microservice runs in its own process (& potentially its own computer in a data center or cloud environment)

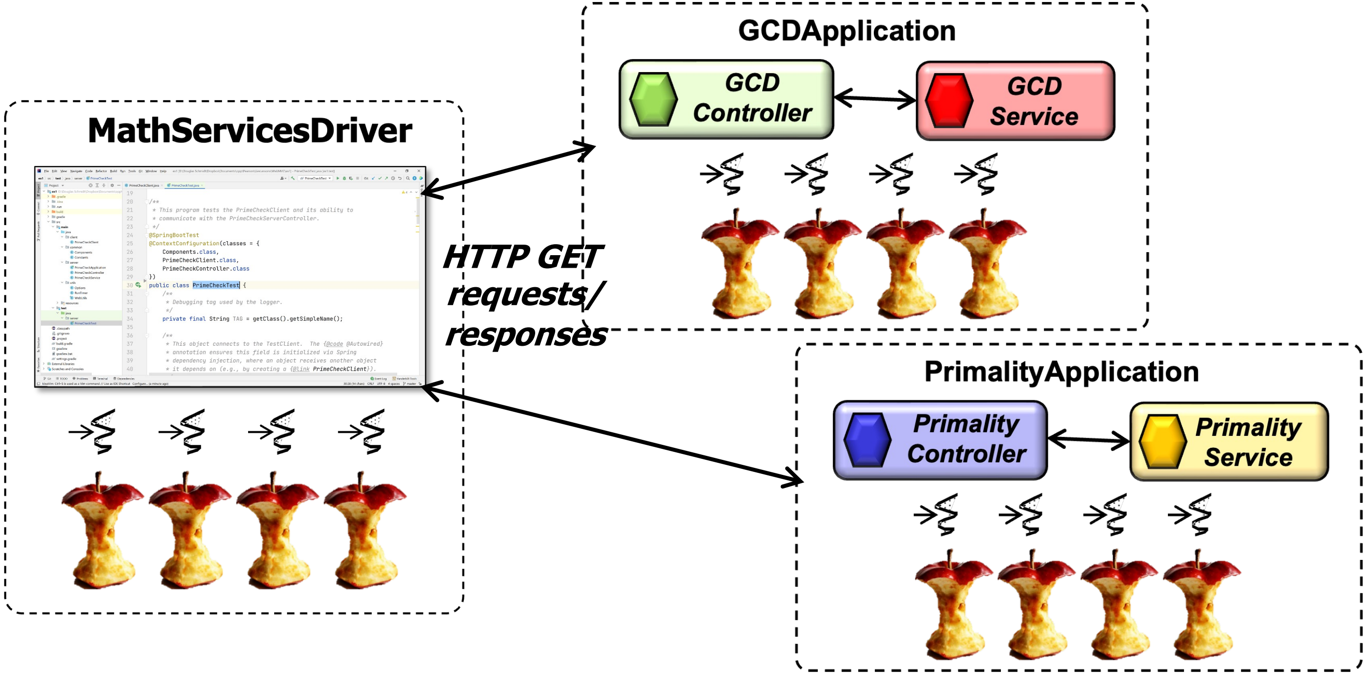


Can improve system scalability & reliability

Pros & Cons of the MathServices App

- Cons
  - Client(s) must be explicitly programmed to connect & communicate with each microservice explicitly





```java
/**
 * This program tests the PrimeCheckClient and its ability to
 * communicate with the PrimeCheckServerController.
 */
@SpringBootTest
@ContextConfiguration(classes = {
    Components.class,
    PrimeCheckClient.class,
    PrimeCheckController.class
})
public class PrimeCheckTest {
    /**
     * Debugging tag used by the logger.
     */
    private final String TAG = getClass().getSimpleName();

    /**
     * This object connects to the TestClient.  The {@code @Autowired}
     * annotation ensures this field is initialized via Spring
     * dependency injection, where an object receives another object
     * it depends on (e.g., by creating a {@link PrimeCheckClient}).
```

Complicates configuration, deployment, testing, & security

End of the MathServices App Case Study: Overview