The MathServices App Case Study: GCD Microservice Structure & Functionality

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

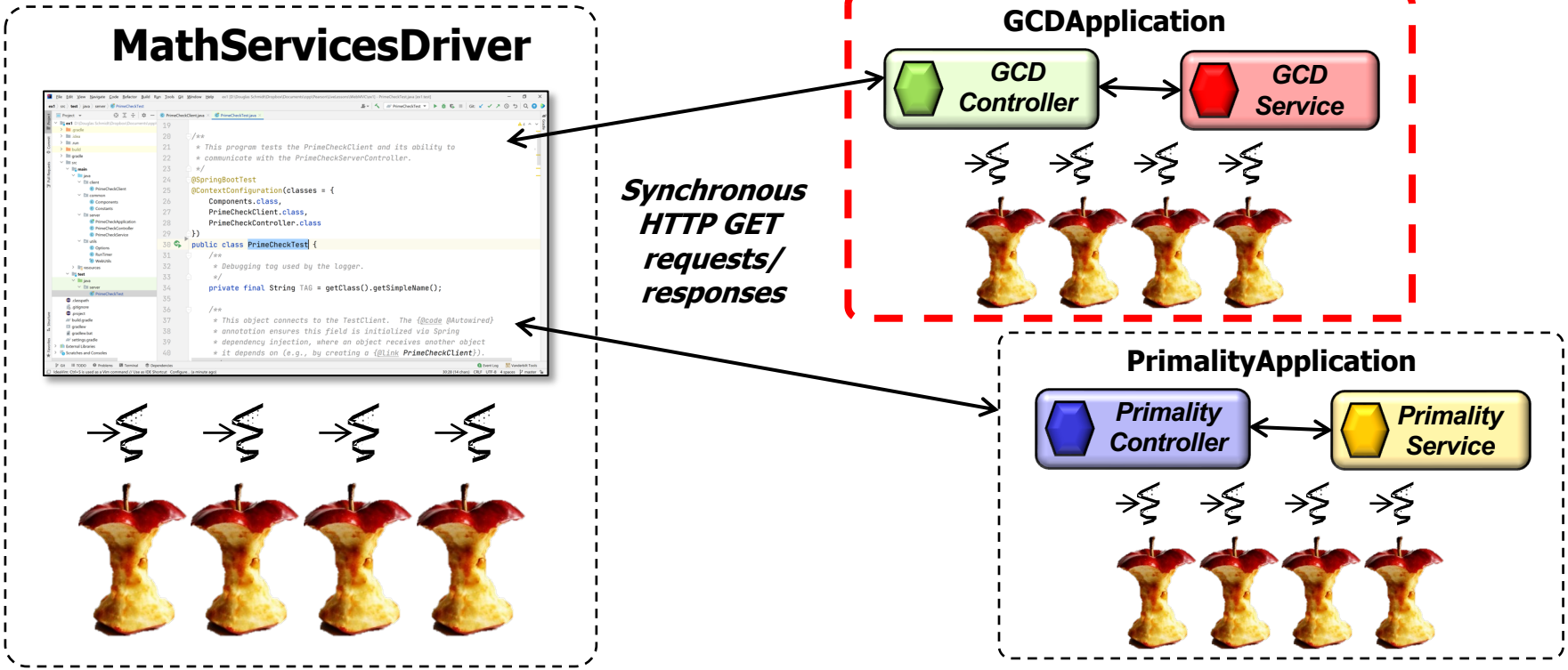
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the structure & functionality of the GCDController/GCDService microservice implementation & how it applies Java parallel streams



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

Structure & Functionality of the GCDController

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

```
@RestController
public class GCDController {
  @Autowired
  GCDService mService;

  @GetMapping(COMPUTE_GCD_LIST)
  public List<GCDResult>
  computeGCDs
    (@RequestParam List<Integer>
       integers) {
    mService
      .computeGCDs(integers);
  }
}
```

See mathservices/microservices/gcd/GCDController.java

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

This annotation ensures request handling methods in the controller class automatically serialize return objects into HttpResponse objects

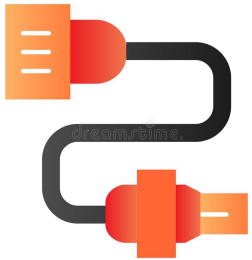
```
@RestController
public class GCDController {
   @Autowired
   GCDService mService;

   @GetMapping(COMPUTE_GCD_LIST)
   public List<GCDResult>
   computeGCDs
      (@RequestParam List<Integer>
         integers) {
      mService
         .computeGCDs(integers);
   }
}
```

See www.baeldung.com/spring-controller-vs-restcontroller

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class



This field is auto-wired by Spring's dependency injection framework

```
@RestController
public class GCDController {
  @Autowired
  GCDService mService;

  @GetMapping(COMPUTE_GCD_LIST)
  public List<GCDResult>
  computeGCDs
    (@RequestParam List<Integer>
       integers) {
    mService
      .computeGCDs(integers);
  }
}
```

See www.baeldung.com/spring-autowire

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

```
@RestController
public class GCDController {
  @Autowired
  GCDService mService;

  @GetMapping(COMPUTE_GCD_LIST)
  public List<GCDResult>
  computeGCDs
    (@RequestParam List<Integer>
       integers) {
    mService
      .computeGCDs(integers);
  }
}
```

This method just forwards to the GCDService method & returns the results back

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

```
@RestController
public class GCDController {
   @Autowired
   GCDService mService;

   @GetMapping(COMPUTE_GCD_LIST)
   public List<GCDResult>
   computeGCDs
      (@RequestParam List<Integer>
         integers) {
      mService
         .computeGCDs(integers);
   }
}
```

This annotation maps HTTP GET requests onto endpoint handler methods

See www.baeldung.com/spring-new-requestmapping-shortcuts

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

This string is used to automatically identify the endpoint handler methods from incoming GET requests

```
@RestController
public class GCDController {
  @Autowired
  GCDService mService;

  @GetMapping(COMPUTE_GCD_LIST)
  public List<GCDResult>
  computeGCDs
    (@RequestParam List<Integer>
       integers) {
    mService
      .computeGCDs(integers);
  }
}
```

Structure & Functionality of the GCDController

- Client HTTP GET requests are mapped to endpoint handler methods via the GCDController class

```
@RestController
public class GCDController {
  @Autowired
  GCDService mService;

  @GetMapping(COMPUTE_GCD_LIST)
  public List<GCDResult>
  computeGCDs
    (@RequestParam List<Integer>
       integers) {
    mService
      .computeGCDs(integers);
  }
}
```

This annotation maps to query parameters, form data, & parts in multipart requests

See www.baeldung.com/spring-request-param

Structure & Functionality of the GCDService

Structure & Functionality of the GCDService

- The GCDService class defines implementation methods that are called by the GCDController

```
@Service
public class GCDService
  public List<GCDResult> computeGCDs
    (List<Integer> integers) {
    return StreamSupport
      .stream(new ListSpliterator(integers),
              true)

      .map(GCDService::computeGCD)

      .toList();
    }
  } ...
```

See mathservices/microservices/gcd/GCDService.java

Structure & Functionality of the GCDService

- The GCDService class defines implementation methods that are called by the GCDController

```
@Service
public class GCDService
  public List<GCDResult> computeGCDs
    (List<Integer> integers) {
   return StreamSupport
     .stream(new ListSpliterator(integers),
             true)

     .map(GCDService::computeGCD)

     .toList();
   }
 } ...
```

This annotation indicates the class implements "business logic" & enables auto-detection & wiring of dependent classes via classpath scanning

See www.baeldung.com/spring-component-repository-service

Structure & Functionality of the GCDService

- The GCDService class defines implementation methods that are called by the GCDController

```
@Service
public class GCDService
  public List<GCDResult> computeGCDs
    (List<Integer> integers) {
    return StreamSupport
      .stream(new ListSpliterator(integers),
              true)

      .map(GCDService::computeGCD)

      .toList();
    }
  } ...
```

Concurrently compute the GCD of the integers param & return GCDResult objects

Structure & Functionality of the GCDService

- The GCDService class defines implementation methods that are called by the GCDController

```
@Service
public class GCDService
  public List<GCDResult> computeGCDs
    (List<Integer> integers) {
    return StreamSupport
      .stream(new ListSpliterator(integers),
              true)

      .map(GCDService::computeGCD)

      .toList();
    }
  } ...
```

Use Java parallel streams to perform all the GCD computations in parallel

See docs.oracle.com/javase/tutorial/collections/streams/parallelism.html

End of the MathServices App Case Study: GCD MicroService Structure & Functionality

The MathServices App Case Study: Implementing the GCD Microservice

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

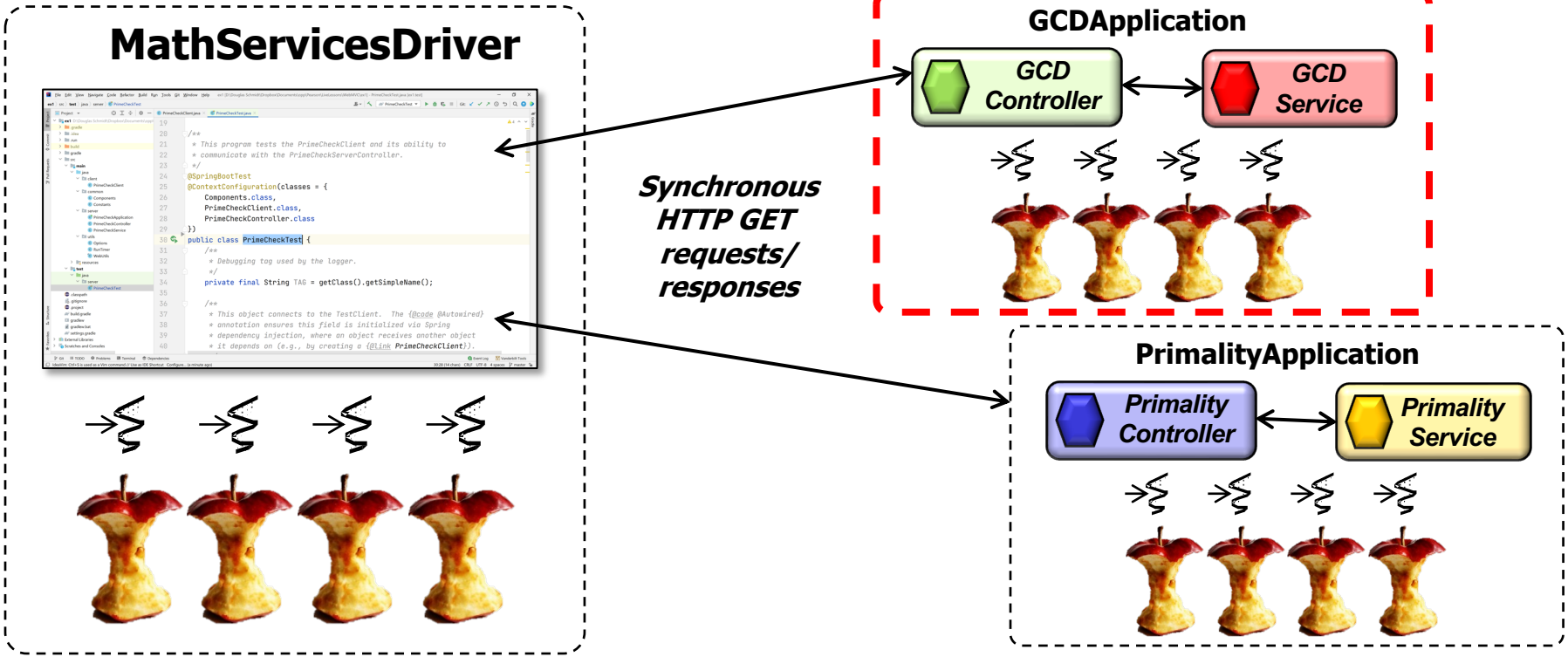
Institute for Software Integrated Systems

Vanderbilt University

Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

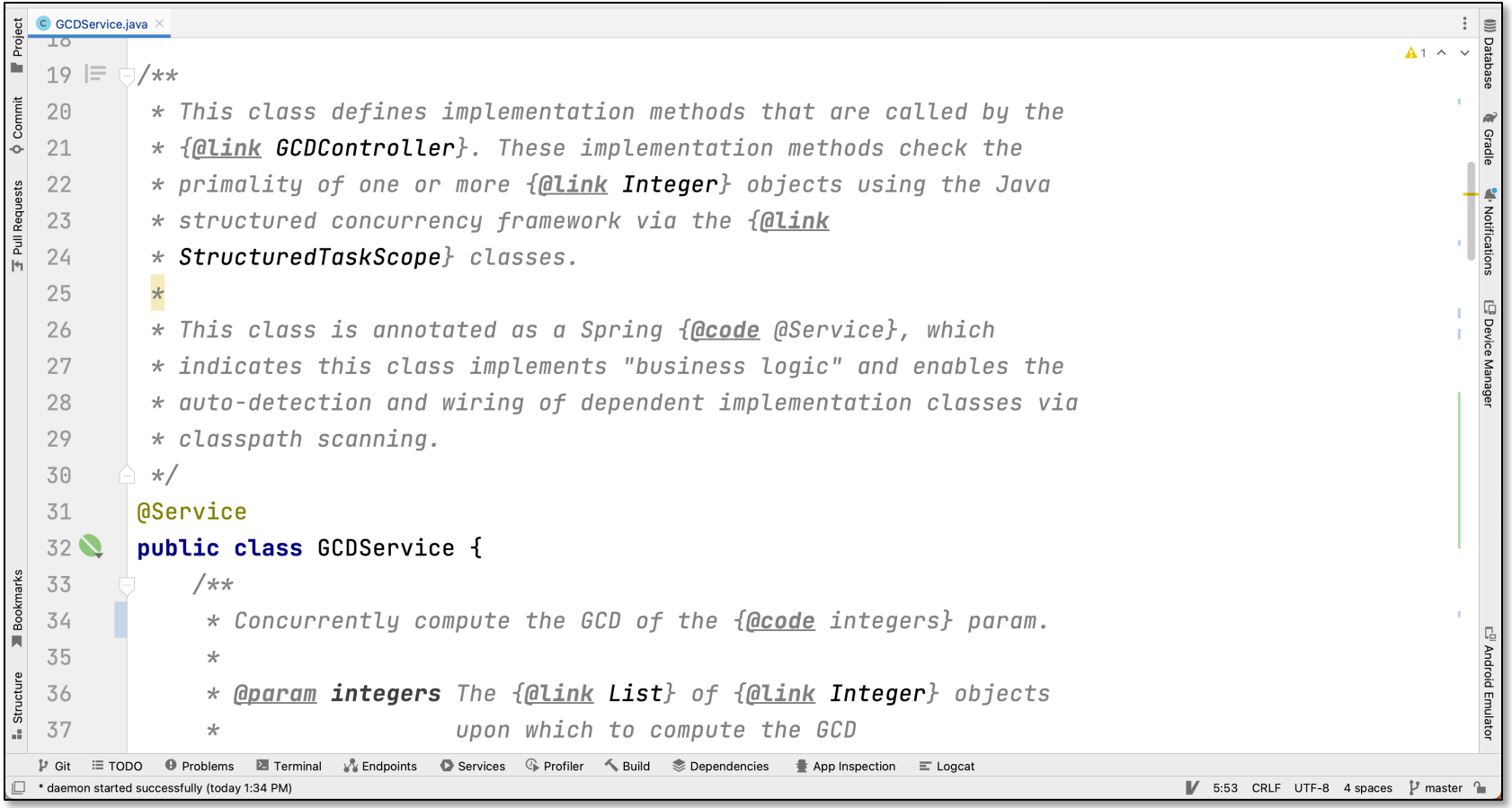
- Understand the concurrent implementation of the GCDController & GCDService classes that run in the GCDApplication microservice



The focus is on the Java parallel streams framework

Implementing the GCDApplication Microservice

Implementing the GCDApplication Microservice



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

End of the MathServices App Case Study: Implementing the GCD Microservices