The MathServices App Case Study: Client Structure & Functionality

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

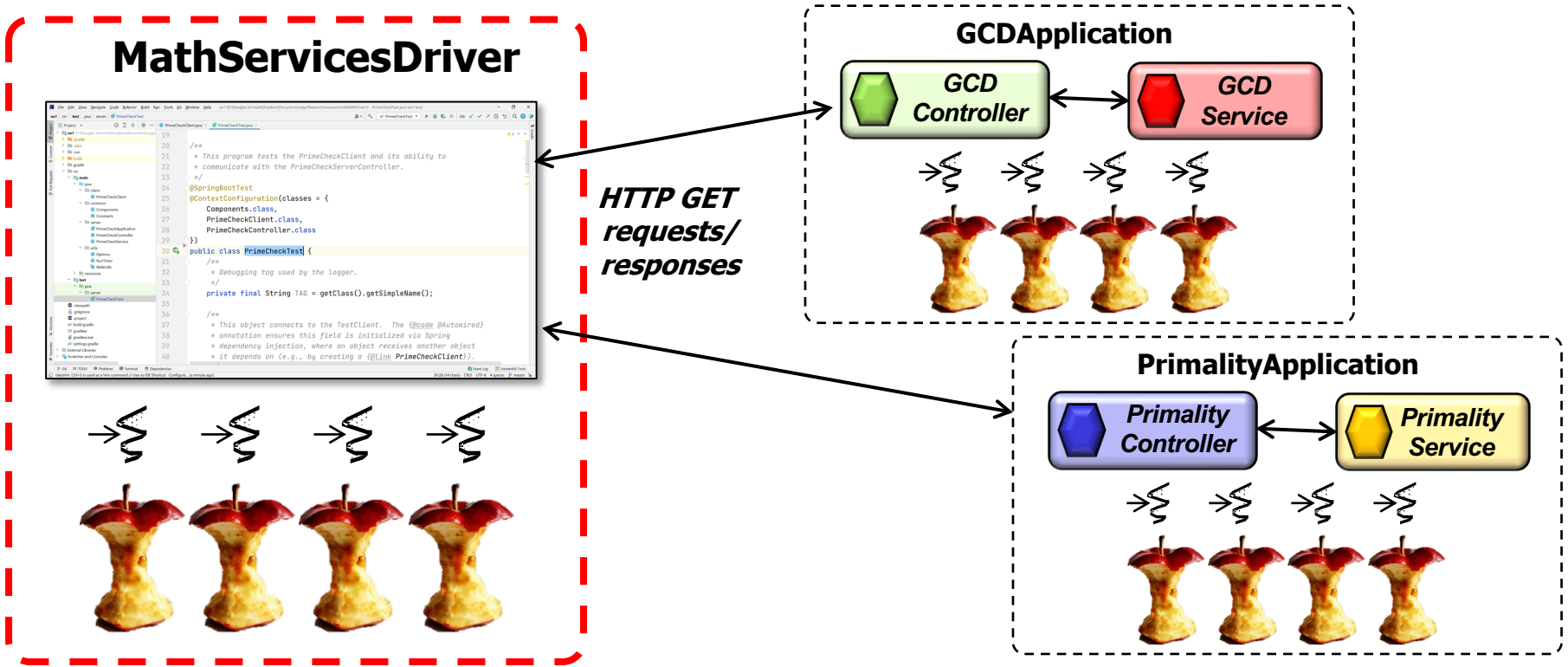
Institute for Software Integrated Systems

Vanderbilt University

Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the structure & functionality of client-related classes that send /receive HTTP GET requests/responses to/from MathServices microservices



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

The Structure & Functionality of MathServicesClient Class

The Structure & Functionality of MathServicesClient Class

- The MathServicesClient class performs synchronous remote method invocations on the microservices to perform math operations concurrently

```
@Component
public class MathServicesClient
  @Autowired GCDProxy mGCDProxy;
  @Autowired PrimalityProxy mPrimalityProxy;

  public List<PrimeResult> checkPrimalities
    (List<Integer> primeCandidates)
    { mPrimalityProxy.checkPrimalities(primeCandidates); }


  public List<GCDResult> computeGCDs(List<Integer> integers)
    { mGCDProxy.computeGCDs(integers); }
}
```

See mathservices/client/MathServicesClient.java

The Structure & Functionality of MathServicesClient Class

- The MathServicesClient class performs synchronous remote method invocations on the microservices to perform math operations concurrently

```
@Component
public class MathServicesClie
  @Autowired GCDProxy mGCDPr
  @Autowired PrimalityProxy mPrimalityProxy;

  public List<PrimeResult> checkPrimalities
    (List<Integer> primeCandidates)
    { mPrimalityProxy.checkPrimalities(primeCandidates); }


  public List<GCDResult> computeGCDs(List<Integer> integers)
    { mGCDProxy.computeGCDs(integers); }
}
```

Enables auto-detection & wiring of dependent implementation classes via classpath scanning

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

The Structure & Functionality of MathServicesClient Class

- The MathServicesClient class performs synchronous remote method invocations on the microservices to perform math operations concurrently

```
@Component
public class MathServicesClient
  @Autowired GCDProxy mGCDProxy;
  @Autowired PrimalityProxy mPrimalityProxy;

  public List<PrimeResult> checkPrimalities
    (List<Integer> primeCandidates)
    { mPrimalityProxy.checkPrimalities(primeCandidates); }


  public List<GCDResult> computeGCDs(List<Integer> integers)
    { mGCDProxy.computeGCDs(integers); }
}
```

Spring's dependency injection framework auto-wires these fields

See www.baeldung.com/spring-autowire

The Structure & Functionality of MathServicesClient Class

- The MathServicesClient classes performs synchronous remote method invocations on the microservices to perform math operations concurrently

```
@Component
public class MathServicesClient
   @Autowired GCDProxy mGCDProxy;
   @Autowired PrimalityProxy mPrimalityProxy;

   public List<PrimeResult> checkPrimalities
      (List<Integer> primeCandidates)
      { mPrimalityProxy.checkPrimalities(primeCandidates); }


   public List<GCDResult> computeGCDs(List<Integer> integers)
      { mGCDProxy.computeGCDs(integers); }
}
```

Sends a List of Integer objects in one HTTP GET request to each microservice

The Structure & Functionality of the *Proxy Classes

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
   @Autowired RestTemplate mRestTemplate;
   ...
```

See WebMVC/ex3/client/src/main/java/edu/vandy/mathservices/client/PrimalityProxy.java

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

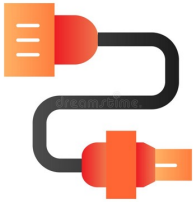
```
@Component
public class PrimalityProxy {
   @Autowired RestTemplate mRestTemplate;
   ...
```

This annotation enables the auto-detection & wiring of dependent implementation classes via classpath scanning

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
   @Autowired RestTemplate mRestTemplate;
   ...
```



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

The assocated @Bean factory method is implemented differently than before

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy { ...
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

This proxy method shields clients from low-level HTTP programming details

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy { ...
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

Create a URI passed in an HTTP GET request to determine if an Integer is prime

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy { ...
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

Set the protocol, port number, & host address

e.g., http://localhost:8082/

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

Set the route name in the path

e.g., checkPrimalityList

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

Convert the List of Integers into a String of comma-separated integers encodings

e.g., "218315,42673259,212438568,147483,5489341,81931857,..."

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    var uri = UriComponentsBuilder
      .newInstance()
      .scheme("http")
      .port(PRIMALITY_MICROSERVICE_PORT)
      .host(HOST)
      .path(CHECK_PRIMALITY_LIST)
      .queryParam("primeCandidates", WebUtils
                  .list2String(primeCandidates))
      .build()
      .toUriString(); ...
```

Build the URI String

e.g., http://localhost:8082/checkPrimalityList?...primeCandidates=218315,147483,...

The Structure & Functionality of the *Proxy Classes

- PrimalityProxy abstracts details of remote method invocations using HTTP

```
@Component
public class PrimalityProxy {
  public List<Integer> checkPrimalities
    (List<Integer> primeCandidates) {
    ...
    return WebUtils
       .makeGetRequestList(mRestTemplate,
                           uri,
                           Integer[].class);
  }
```

Make an HTTP GET call to the server at the designed URL to check the primalities in the List

e.g., http://localhost:8082/checkPrimalityList?...primeCandidates=218315,147483,...

The Structure & Functionality of the *Proxy Classes

- Likewise, GCDProxy abstracts details of remote method invocations using HTTP

```
@Component
public class GCDProxy {
  public List<GCDResult> computeGCDs(List<Integer> integers) {
    var uri = UriComponentsBuilder.newInstance()
      .scheme("http")
      .port(GCD_MICROSERVICE_PORT)
      .host(HOST)
      .path(COMPUTE_GCD_LIST)
      .queryParam("integers", WebUtils.list2String(integers))
    .build()
    .toUriString();
    return WebUtils
      .makeGetRequestList(mRestTemplate, uri, Integer[].class);
  }
```

Compute the GCD of the integers param

The implementation is similar, except for the port number & path name portions

End of the MathServices App Case Study: Client Structure & Functionality