

The MathServices App Case Study: Implementing the Primality Microservice

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

**Institute for Software
Integrated Systems**

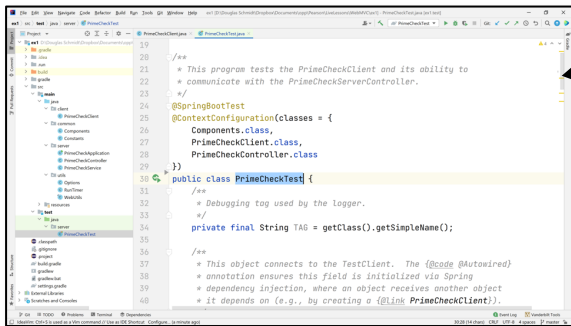
**Vanderbilt University
Nashville, Tennessee, USA**



Learning Objectives in this Part of the Lesson

- Understand the concurrent implementation of the PrimalityController & PrimalityService classes that run in the PrimalityApplication microservice

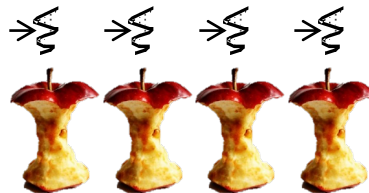
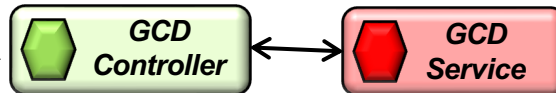
MathServicesDriver



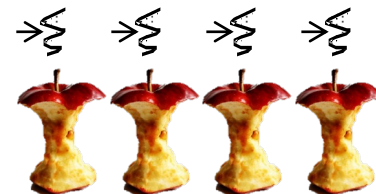
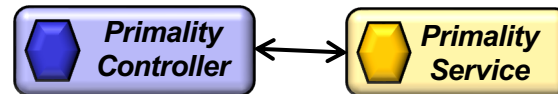
```
19 //**
20 // This program tests the PrimeCheckClient and its ability to
21 // communicate with the PrimeCheckServerController.
22 //**
23 //
24 //
25 @SpringBootTest
26 @ContextConfiguration(classes = {
27     Components.class,
28     PrimeCheckClient.class,
29     PrimeCheckController.class
30 })
31 public class PrimeCheckTest {
32     //**
33     // Debugging tag used by the logger.
34     //**
35     private final String TAG = getClass().getSimpleName();
36
37     //**
38     // This object connects to the TestClient. The @Code @Autowired
39     // annotation ensures this field is initialized via Spring
40     // dependency injection, where an object receives another object
41     // it depends on (e.g., by creating a @Link PrimeCheckClient?).
42 }
```

*HTTP GET
requests/
responses*

GCDApplication



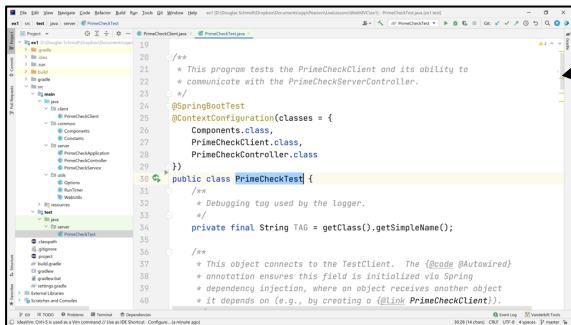
PrimalityApplication



Learning Objectives in this Part of the Lesson

- Understand the implementation of the PrimalityController & PrimalityService classes that run in the PrimalityApplication microservice

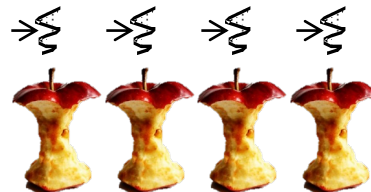
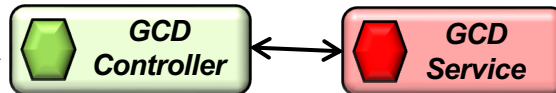
MathServicesDriver



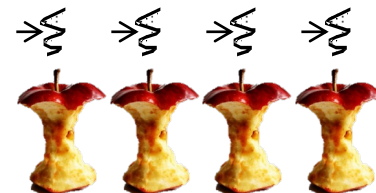
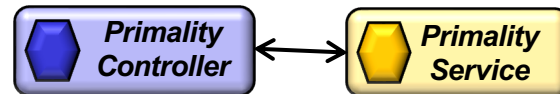
```
20 /**  
21  * This program tests the PrimeCheckClient and its ability to  
22  * communicate with the PrimeCheckServerController.  
23  */  
24 @SpringBootTest  
25 @ContextConfiguration(classes = {  
26     Components.class,  
27     PrimeCheckClient.class,  
28     PrimeCheckController.class  
29 })  
30 public class PrimeCheckTest {  
31     /**  
32      * Debugging tag used by the logger.  
33      */  
34     private final String TAG = getClass().getSimpleName();  
35  
36     /**  
37      * This object connects to the TestClient. The @Autowired  
38      * annotation ensures this field is initialized via Spring  
39      * dependency injection, where an object receives another object  
40      * it depends on (e.g., by creating a @Link PrimeCheckClient).  
41      */  
42      */
```

*HTTP GET
requests/
responses*

GCDApplication



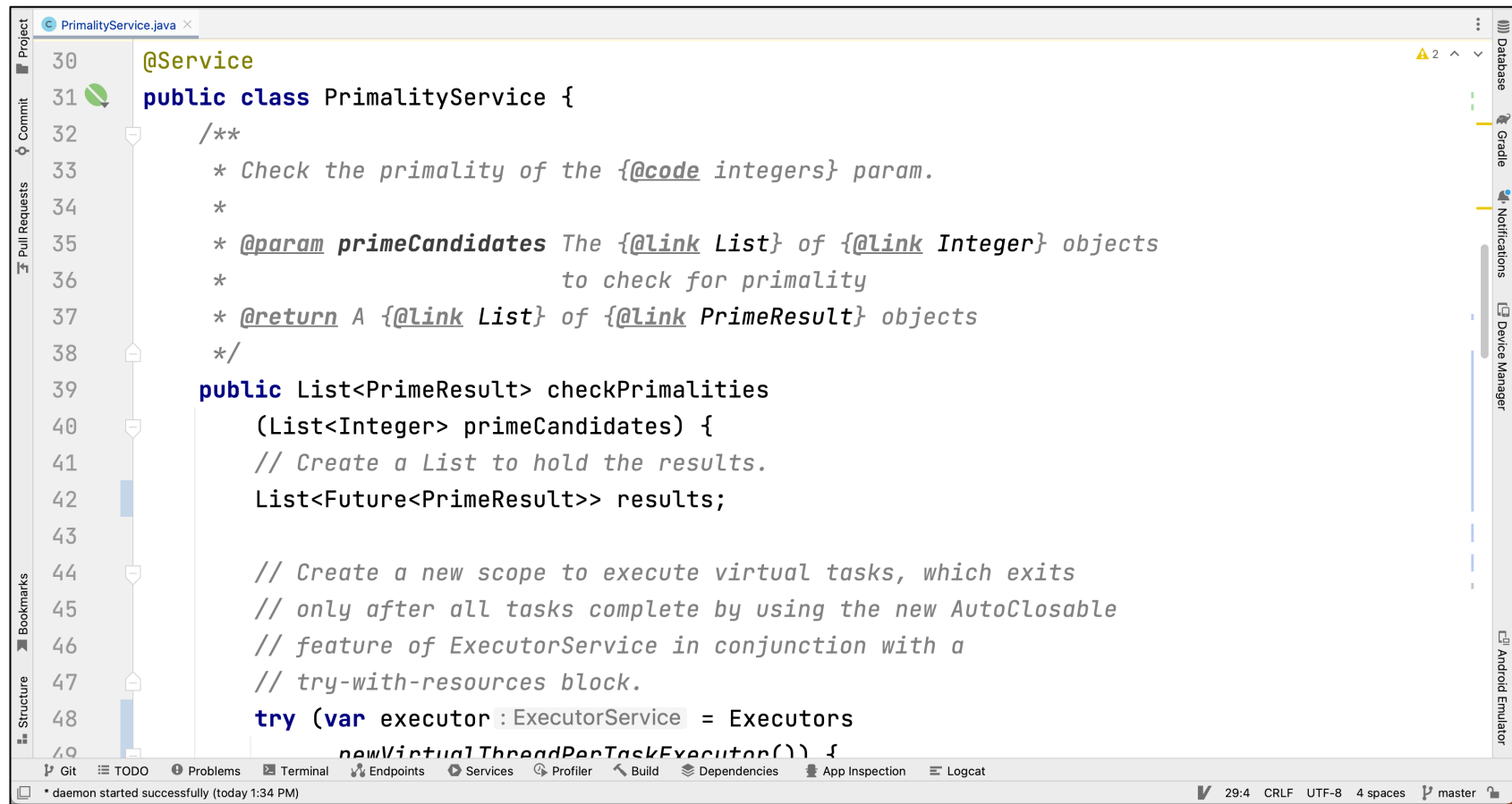
PrimalityApplication



The focus is on the Java Executors VirtualThreadPerTaskExecutor model

Implementing the Primality Application Microservice

Implementing the PrimalityApplication Microservice



The screenshot shows an IDE window with a file named `PrimalityService.java`. The code is as follows:

```
30  @Service
31  public class PrimalityService {
32      /**
33       * Check the primality of the {@code integers} param.
34       *
35       * @param primeCandidates The {@link List} of {@link Integer} objects
36       *                        to check for primality
37       * @return A {@link List} of {@link PrimeResult} objects
38       */
39      public List<PrimeResult> checkPrimalities
40          (List<Integer> primeCandidates) {
41          // Create a List to hold the results.
42          List<Future<PrimeResult>> results;
43
44          // Create a new scope to execute virtual tasks, which exits
45          // only after all tasks complete by using the new AutoClosable
46          // feature of ExecutorService in conjunction with a
47          // try-with-resources block.
48          try (var executor : ExecutorService = Executors
49              newVirtualThreadPerTaskExecutor()) {
```

The IDE interface includes a left sidebar with icons for Project, Commit, Pull Requests, Bookmarks, and Structure. The right sidebar contains icons for Database, Gradle, Notifications, Device Manager, and Android Emulator. The bottom status bar shows the file encoding as UTF-8, 4 spaces, and the master branch. A message at the bottom left states: '* daemon started successfully (today 1:34 PM)'.

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

End of the MathServices App Case Study: Implementing the Primality Microservice