

# The PrimeCheck App Case Study: Implementing the Client

**Douglas C. Schmidt**

**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**

**[www.dre.vanderbilt.edu/~schmidt](http://www.dre.vanderbilt.edu/~schmidt)**

**Professor of Computer Science**

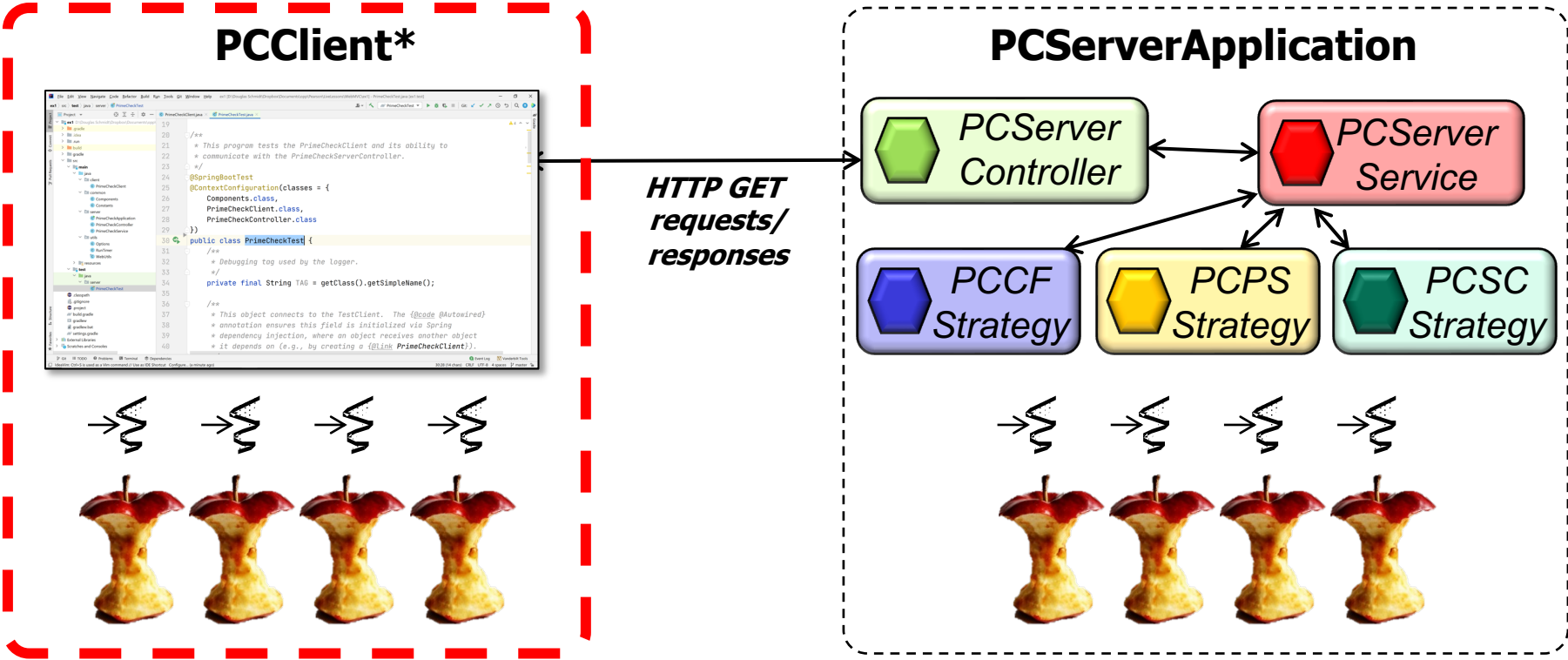
**Institute for Software  
Integrated Systems**

**Vanderbilt University  
Nashville, Tennessee, USA**



# Learning Objectives in this Part of the Lesson

- Understand the PCClient\* class implementations that send/receive HTTP GET requests/responses to/from the PCServiceApplication synchronously



---

# Implementing the PrimeCheck App Client

# Implementing the PrimeCheckApp Client

The screenshot shows an IDE window titled "ex1 - PCClientParallelStream.java [ex1.test]". The left sidebar displays a project structure with the following folders and files:

- main
  - java
    - primechecker
      - common
        - Components
        - Constants
        - Options
      - server
        - strategies
          - PCServerApplication
          - PCServerController
          - PCServerService
      - utils
        - FuturesCollector
        - FutureUtils
        - PrimeUtils
        - RunTimer
      - resources
        - application.properties
    - test
      - java
        - primechecker
          - client
            - PCClientCompletableFuture
            - PCClientParallelStream
            - PCClientStructuredConcurrency
            - PCProxy
          - utils
            - PrimeCheckTest

The main editor area shows the following code:

```
24 @Component
25 public class PCClientParallelStream {
26     /**
27      * This auto-wired field connects the {@link PCClientParallelStream}
28      * the {@link PCProxy} that performs HTTP requests
29      * synchronously.
30      */
31     @Autowired
32     private PCProxy mPCProxy;
33
34     /**
35      * Send individual HTTP GET requests to the server to check if a
36      * the {@code primeCandidates} {@link List} of {@link Integer}
37      * objects are prime or not.
38      *
39      * @param primeCandidates A {@link List} of {@link Integer}
40      *                        objects to check for primality
41      * @param parallel True if using parallel streams, else false
42      * @return A {@link List} of {@link Integer} objects indicating
```

See [github.com/douglasraigschmidt/LiveLessons/tree/master/WebMVC/ex1](https://github.com/douglasraigschmidt/LiveLessons/tree/master/WebMVC/ex1)

---

# End of the PrimeCheck App Case Study: Implementing the Client