

Scouting App Training

Team 294

Introduction to Angular and Web Service Programming

Part 1: Web Services

10/22/2020

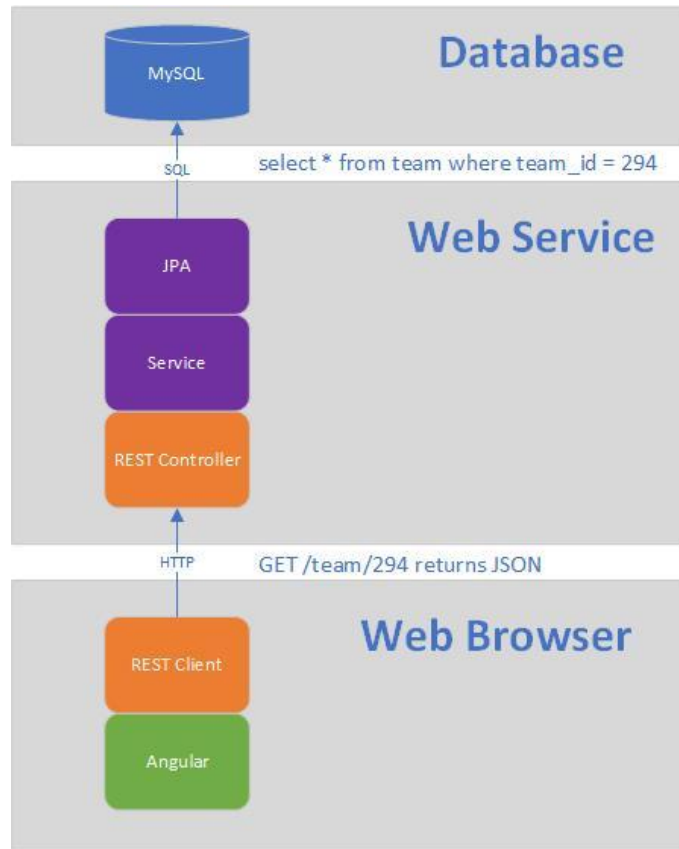
Introduction

The scouting app runs on a server in Amazon

- Ubuntu Linux
- MySQL database
- Java web service
- Angular web application

We are going to start by learning some web service basics but spend most of our time learning web development using Angular.

<https://github.com/team294/OrcaTrain.git>



Code

- Robot has about 5,000 lines of Java code
- Scouting app has about 10,000 lines of code
 - 4,000 lines of Java code
 - 4,000 lines of TypeScript
 - 2,000 lines of HTML and CSS

Technology

Back end

- Java
- SpringBoot
- Java Persistence API (JPA)
- SQL
- Linux
- AWS EC2

Front end

- Angular
- Node.js
- JavaScript / TypeScript
- HTML
- CSS
- Bootstrap

Web Service

The scouting app uses the Spring framework to build a RESTful web service

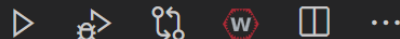
<https://spring.io/>

- SpringBootApplication – run this to start the web service
- RestController – maps the HTTP requests to methods
- Service – component that contains the core logic of the service

Web services are a popular way to build APIs and allow you to break up applications into reusable services that can be called by any modern language

Service

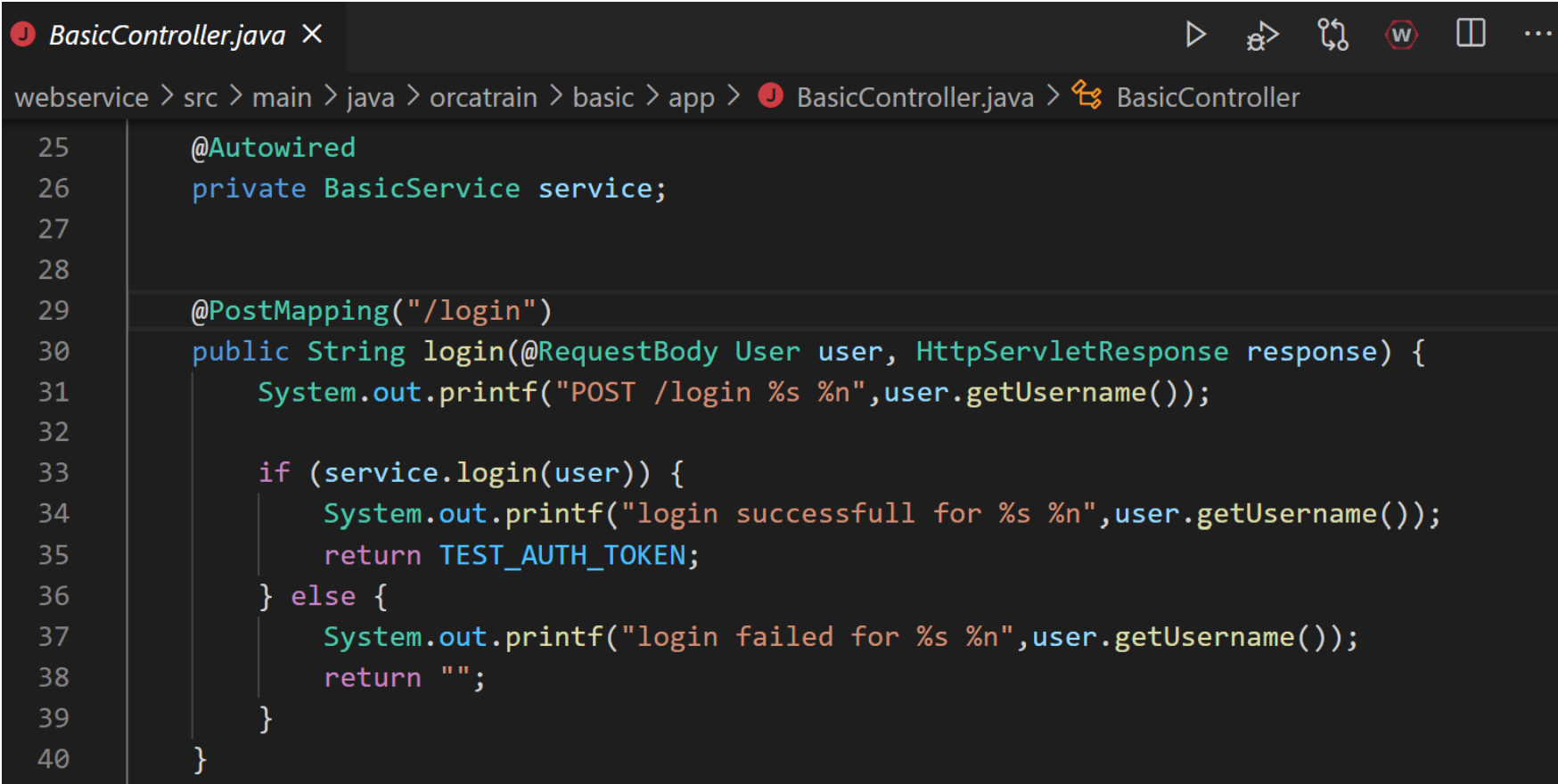
BasicService.java



webservice > src > main > java > orcatrain > basic > app > BasicService.java > BasicService > getUserByName(String)

```
9
10 @Component
11 public class BasicService {
12
13     private List<User> userList = null;
14
15     public User getUserByName(String name) {
16         User found = null;
17         List<User> userList = getUsers();
18         for (User u:userList) {
19             if (u.getUsername().equalsIgnoreCase(name)) {
20                 found = u;
21                 break;
22             }
23         }
24         return found;
25     }
26 }
```

Controller



```
BasicController.java ×
webbservice > src > main > java > orcatrain > basic > app > BasicController.java > BasicController

25  @Autowired
26  private BasicService service;
27
28
29  @PostMapping("/login")
30  public String login(@RequestBody User user, HttpServletResponse response) {
31      System.out.printf("POST /login %s %n",user.getUsername());
32
33      if (service.login(user)) {
34          System.out.printf("login successfull for %s %n",user.getUsername());
35          return TEST_AUTH_TOKEN;
36      } else {
37          System.out.printf("login failed for %s %n",user.getUsername());
38          return "";
39      }
40  }
```

Application

BasicApplication.java X

src > main > java > orcatrain > basic > app > BasicApplication.java > BasicApplication

```
1  package orcatrain.basic.app;
2
3  import org.springframework.boot.SpringApplication;
4  import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6  @SpringBootApplication
7  public class BasicApplication {
8
9      public static void main(String[] args) {
10         SpringApplication.run(BasicApplication.class, args);
11     }
12 }
13
```


Build and Run the Web Service

- gradlew bootjar
- java -jar build/libs/orcatrain-webservice-1.jar

```
C:\Users\Paul\Documents\GitHub\OrcaTrain\webservice\build\libs>java -jar orcatrain-webservice-1.jar
```

```

  ____ _
 / ___| | | |
 \___ \| |_| |
  ___) | | | |
 |_____|_|_|_|

:: Spring Boot ::      (v2.2.2.RELEASE)

```

```
2020-10-20 22:06:50.594 INFO 16920 --- [main] orcatrain.basic.app.BasicApplication : Starting BasicApplication on i726 with PID 16920 (C:\Users\Paul\Documents\GitHub\OrcaTrain\webservice\build\libs\orcatrain-webservice-1.jar started by Paul in C:\Users\Paul\Documents\GitHub\OrcaTrain\webservice\build\libs)
2020-10-20 22:06:50.598 INFO 16920 --- [main] orcatrain.basic.app.BasicApplication : No active profile set, falling back to default profiles: default
2020-10-20 22:06:51.799 INFO 16920 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8090 (http)
2020-10-20 22:06:51.810 INFO 16920 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-20 22:06:51.811 INFO 16920 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.29]
2020-10-20 22:06:51.872 INFO 16920 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-20 22:06:51.872 INFO 16920 --- [main] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 1198 ms
2020-10-20 22:06:52.070 INFO 16920 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-20 22:06:52.238 INFO 16920 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8090 (http) with context path ''
2020-10-20 22:06:52.243 INFO 16920 --- [main] orcatrain.basic.app.BasicApplication : Started BasicApplication in 2.204 seconds (JVM running for 2.74)
```

Postman

<https://www.postman.com/downloads/>

The screenshot displays the Postman application window. The top bar includes the Postman logo, menu items (File, Edit, View, Help), and buttons for '+ New', 'Import', 'Runner', and 'My Workspace'. The left sidebar shows a 'Collections' tab with a search filter, a 'History' tab, and a collection named 'OrcaTrain' containing two requests: 'POST Login' and 'GET Get Users'. The main workspace shows the 'POST Login' request selected. The request details are as follows:

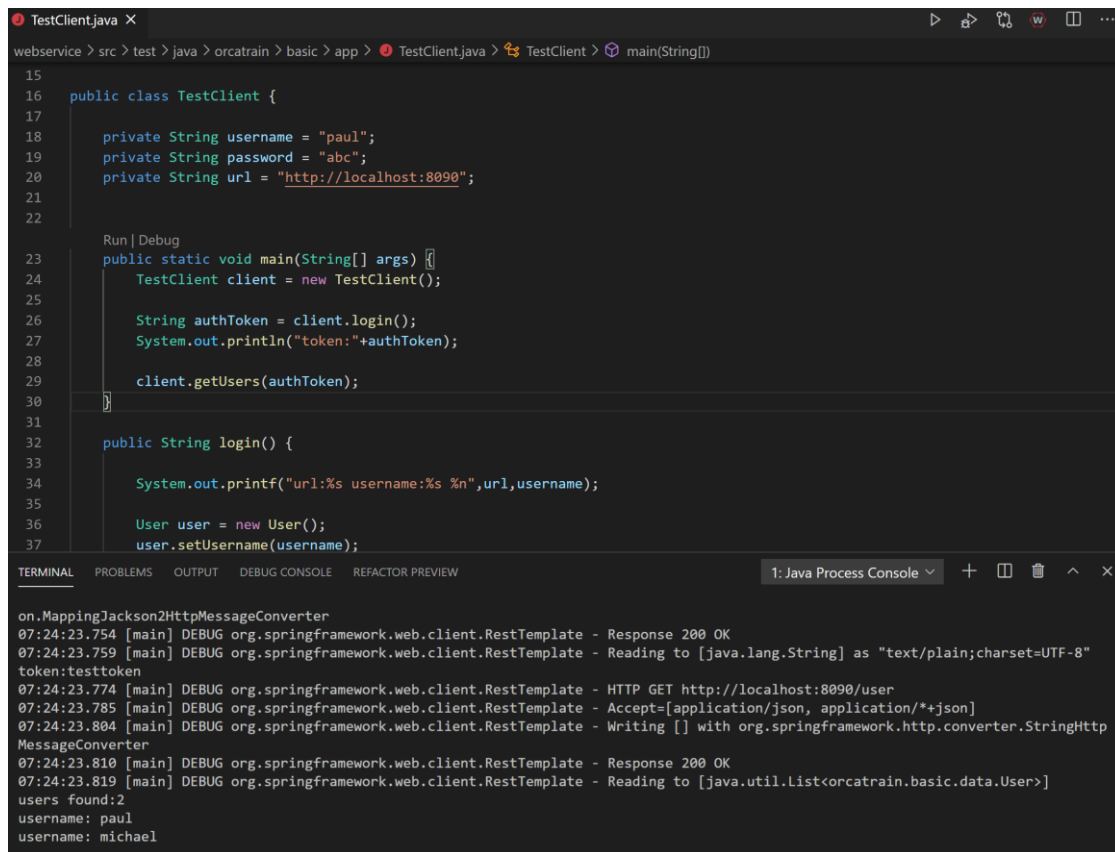
- Method:** POST
- URL:** localhost:8090/login
- Body:** The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "username": "paul",
3   "password": "abc"
4 }
```
- Params:** none
- Authorization:** form-data
- Headers:** 8
- Pre-request Script:** Tests
- Settings:** Settings

The bottom status bar shows the response details: **Status:** 200 OK, **Time:** 15 ms, **Size:** 261 B. The response body is displayed in the 'Body' tab, showing a single line:

```
1 testtoken
```

TestClient



The screenshot shows an IDE window titled "TestClient.java" with a file explorer on the left showing the path: `webservice > src > test > java > orcatrain > basic > app > TestClient.java`. The code in the editor is as follows:

```
15
16 public class TestClient {
17
18     private String username = "paul";
19     private String password = "abc";
20     private String url = "http://localhost:8090";
21
22
23     Run | Debug
24     public static void main(String[] args) {
25         TestClient client = new TestClient();
26
27         String authToken = client.login();
28         System.out.println("token:"+authToken);
29
30         client.getUsers(authToken);
31     }
32
33     public String login() {
34         System.out.printf("url:%s username:%s %n",url,username);
35
36         User user = new User();
37         user.setUsername(username);
```

The terminal at the bottom shows the following output:

```
on.MappingJackson2HttpMessageConverter
07:24:23.754 [main] DEBUG org.springframework.web.client.RestTemplate - Response 200 OK
07:24:23.759 [main] DEBUG org.springframework.web.client.RestTemplate - Reading to [java.lang.String] as "text/plain;charset=UTF-8"
token:testtoken
07:24:23.774 [main] DEBUG org.springframework.web.client.RestTemplate - HTTP GET http://localhost:8090/user
07:24:23.785 [main] DEBUG org.springframework.web.client.RestTemplate - Accept=[application/json, application/*+json]
07:24:23.804 [main] DEBUG org.springframework.web.client.RestTemplate - Writing [] with org.springframework.http.converter.StringHttp
MessageConverter
07:24:23.810 [main] DEBUG org.springframework.web.client.RestTemplate - Response 200 OK
07:24:23.819 [main] DEBUG org.springframework.web.client.RestTemplate - Reading to [java.util.List<orcatrain.basic.data.User>]
users found:2
username: paul
username: michael
```