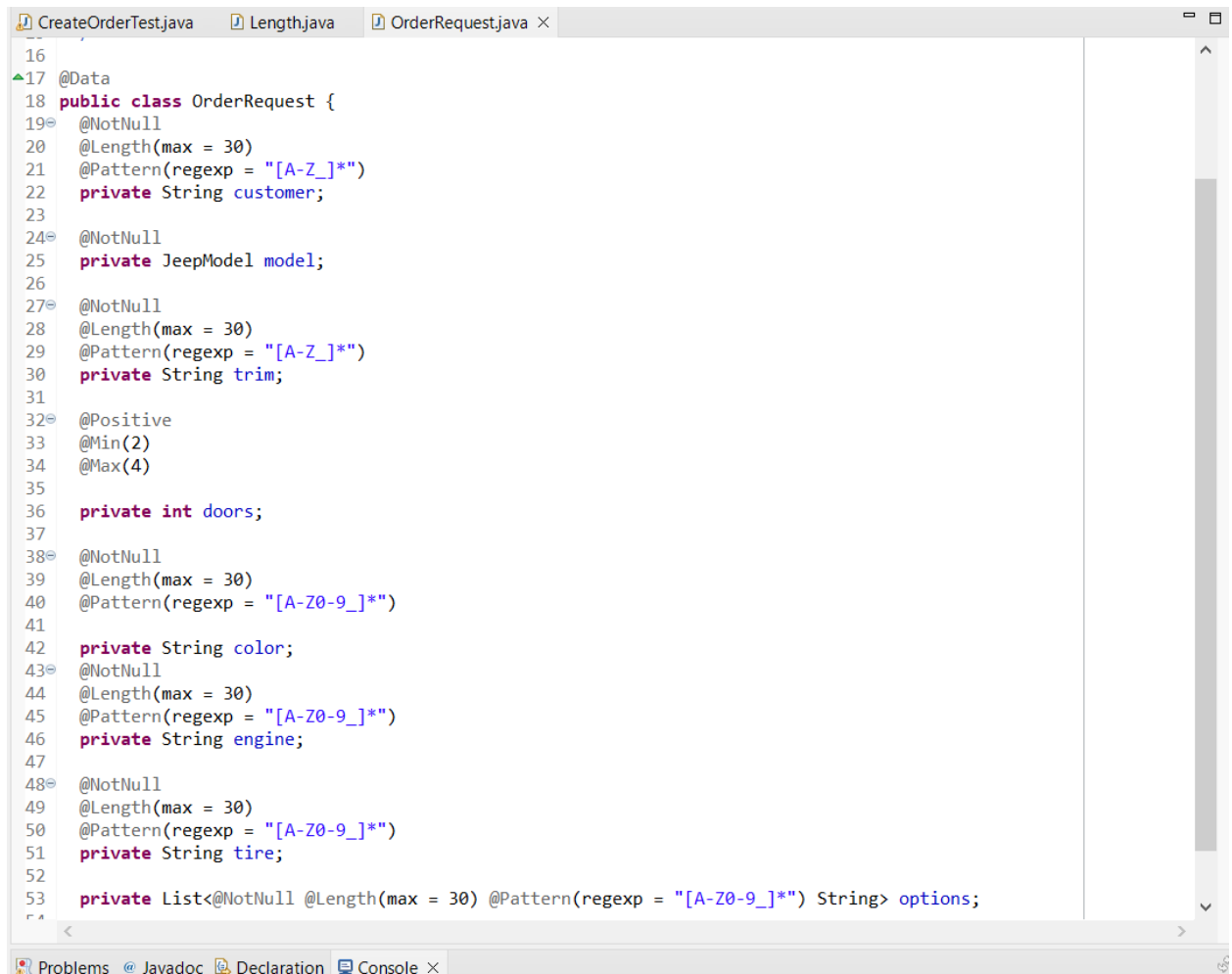
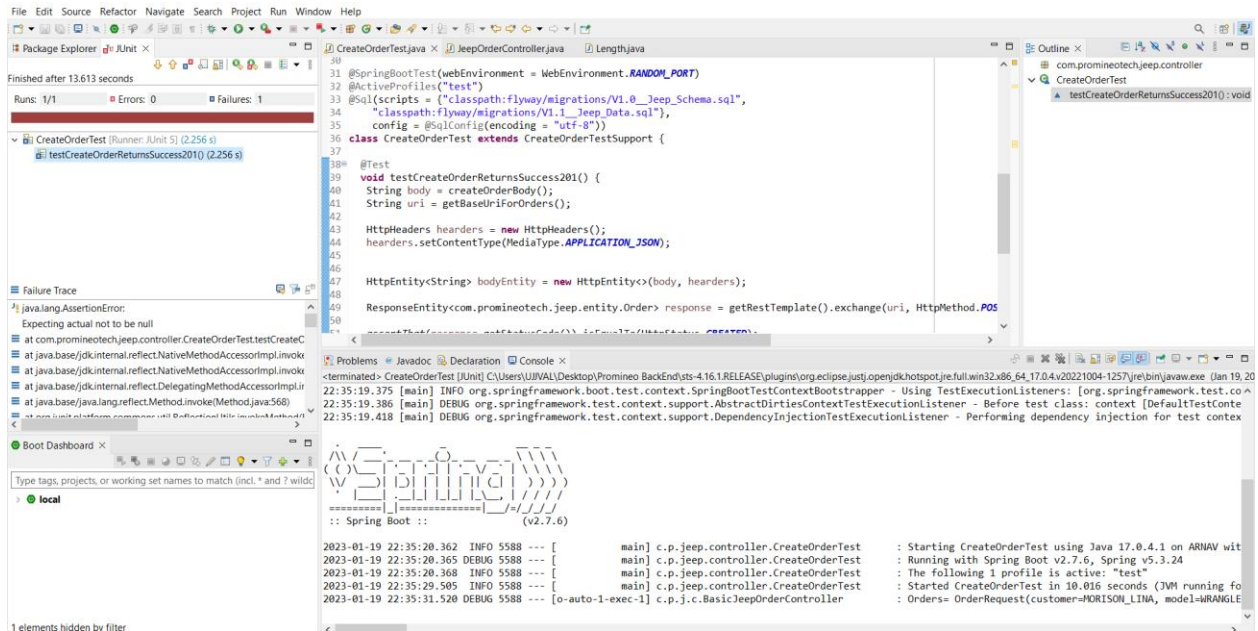


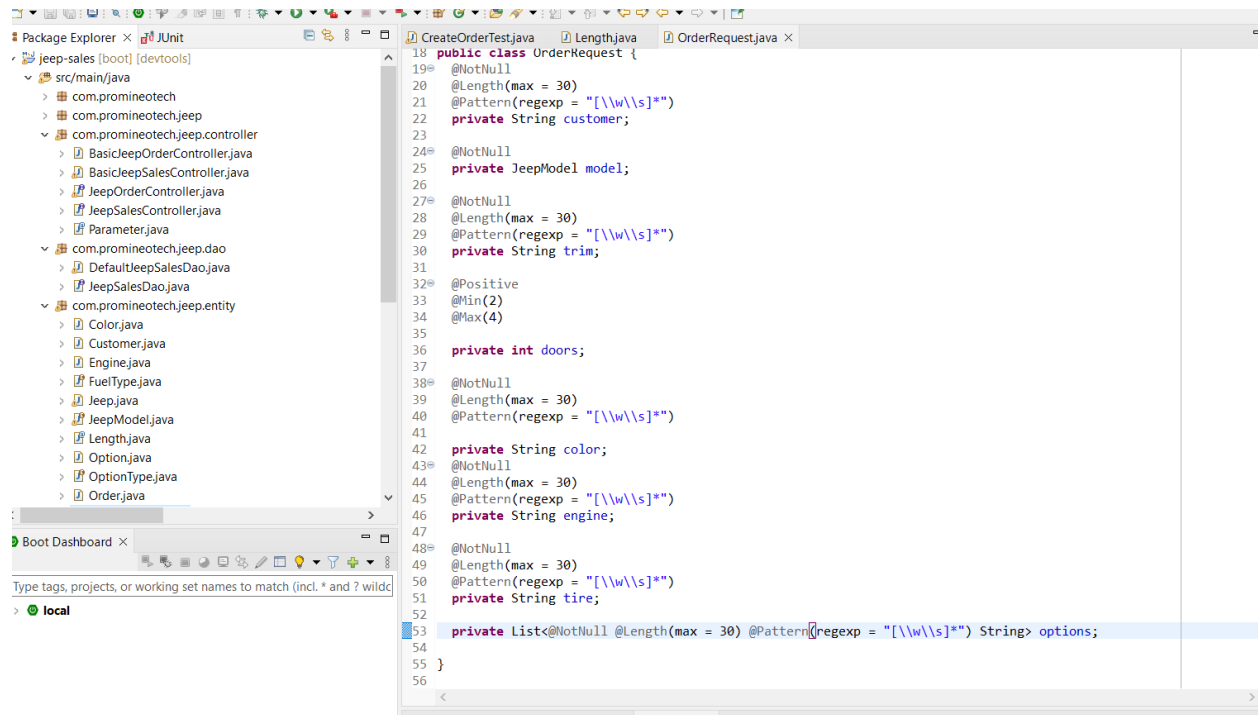
```
CreateOrderTest.java × CreateOrderTestSupport.java
30
31 @SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
32 @ActiveProfiles("test")
33 @Sql(scripts = {"classpath:flyway/migrations/V1.0__Jeep_Schema.sql",
34 "classpath:flyway/migrations/V1.1__Jeep_Data.sql"},
35 config = @SqlConfig(encoding = "utf-8"))
36 class CreateOrderTest extends CreateOrderTestSupport {
37
38     @Test
39     void testCreateOrderReturnsSuccess201() {
40         String body = createOrderBody();
41         String uri = getBaseUrlForOrders();
42
43         HttpHeaders headers = new HttpHeaders();
44         headers.setContentType(MediaType.APPLICATION_JSON);
45
46
47         HttpEntity<String> bodyEntity = new HttpEntity<>(body, headers);
48
49         ResponseEntity<com.promineotech.jeep.entity.Order> response = getRestTemplate().exchange(uri, HttpMethod.POST,
50 bodyEntity, com.promineotech.jeep.entity.Order.class);
51
52         assertThat(response.getStatusCode()).isEqualTo(HttpStatus.CREATED);
53
54         assertThat(response.getBody()).isNotNull();
55
56         com.promineotech.jeep.entity.Order order = response.getBody();
57         assertThat(order.getCustomer().getCustomerId()).isEqualTo("MORISON_LINA");
58         assertThat(order.getModel().getModelId()).isEqualTo(JeepModel.WRANGLER);
59         assertThat(order.getModel().getTrimLevel()).isEqualTo("Sport Altitude");
60         assertThat(order.getModel().getNumDoors()).isEqualTo(4);
61         assertThat(order.getColor().getColorId()).isEqualTo("EXT_NACHO");
62         assertThat(order.getEngine().getEngineId()).isEqualTo("2_0_TURBO");
63         assertThat(order.getTire().getTireId()).isEqualTo("35_TOYO");
64         assertThat(order.getOptions()).hasSize(6);
65     }
66
67 }
68
```

Problems Javadoc Declaration Console ×

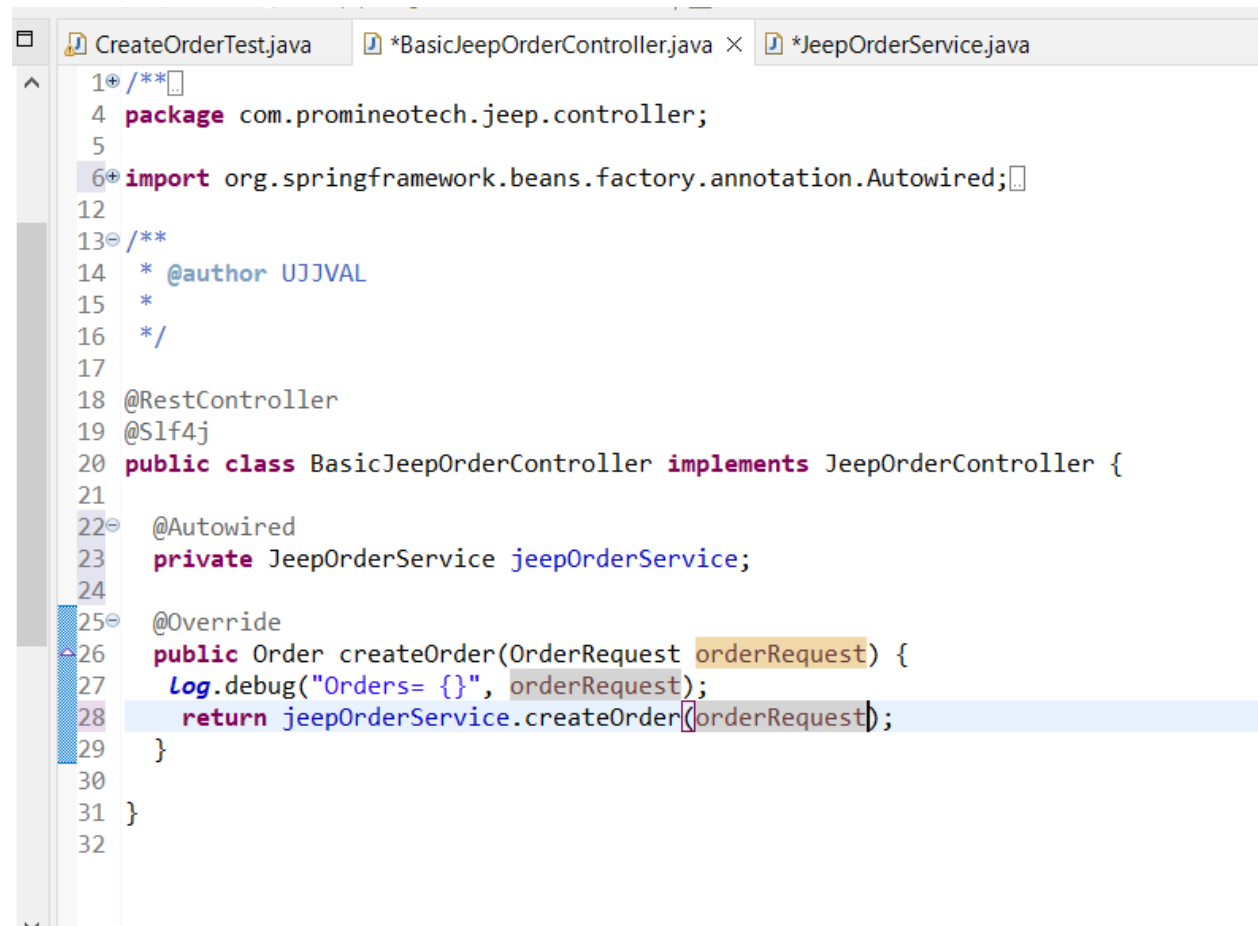
```
CreateOrderTest.java JeepOrderController.java × *Length.java
24 import io.swagger.v3.oas.annotations.servers.Server;
25 @Validated
26 @RequestMapping("/orders")
27 @OpenAPIDefinition(info = @Info(title = "Jeep order Services"), servers = {
28     @Server(url = "http://localhost:8080", description = "Local server.")})
29 public interface JeepOrderController {
30     // @formatter:off
31     @Operation(
32         summary = "Create an order for jeep",
33         description = "Returns Created jeep",
34         responses = {
35             @ApiResponse(responseCode = "201",
36                 description = "The Created jeeps returned",
37                 content = @Content(mediaType = "application/json",
38                     schema = @Schema(implementation = Order.class))),
39             @ApiResponse(responseCode = "400",
40                 description = "request parameters are invalid",
41                 content = @Content(mediaType = "application/json")),
42             @ApiResponse(responseCode = "404",
43                 description = "no jeep component were found with input",
44                 content = @Content(mediaType = "application/json")),
45             @ApiResponse(responseCode = "500",
46                 description = "error",
47                 content = @Content(mediaType = "application/json"))
48         },
49         parameters = {
50             @Parameter(
51                 name = "orderRequest",
52                 required = true,
53                 description = "The order as json"),
54         }
55     )
56 }
57
58 // @formatter:on
59 @PostMapping
60 @ResponseStatus(code = HttpStatus.CREATED)
61 Order createOrder(@Valid @RequestBody OrderRequest orderRequest);
```

Problems Javadoc Declaration Console ×





```
18 public class OrderRequest {
19     @NotNull
20     @Length(max = 30)
21     @Pattern(regexp = "[\\w\\s]*")
22     private String customer;
23
24     @NotNull
25     private JeepModel model;
26
27     @NotNull
28     @Length(max = 30)
29     @Pattern(regexp = "[\\w\\s]*")
30     private String trim;
31
32     @Positive
33     @Min(2)
34     @Max(4)
35     private int doors;
36
37     @NotNull
38     @Length(max = 30)
39     @Pattern(regexp = "[\\w\\s]*")
40     private String color;
41
42     @NotNull
43     @Length(max = 30)
44     @Pattern(regexp = "[\\w\\s]*")
45     private String engine;
46
47     @NotNull
48     @Length(max = 30)
49     @Pattern(regexp = "[\\w\\s]*")
50     private String tire;
51
52     private List<@NotNull @Length(max = 30) @Pattern(regexp = "[\\w\\s]*") String> options;
53
54 }
55
56
```



```
1+ /**
4 package com.promineotech.jeep.controller;
5
6+ import org.springframework.beans.factory.annotation.Autowired;
12
13+ /**
14 * @author UJJVAL
15 *
16 */
17
18 @RestController
19 @Slf4j
20 public class BasicJeepOrderController implements JeepOrderController {
21
22     @Autowired
23     private JeepOrderService jeepOrderService;
24
25     @Override
26     public Order createOrder(OrderRequest orderRequest) {
27         log.debug("Orders= {}", orderRequest);
28         return jeepOrderService.createOrder(orderRequest);
29     }
30
31 }
32
```

```
CreateOrderTest.java BasicJeepOrderController.java DefaultJeepOrderService.java x
1 /**
4 package com.promineotech.jeeo.service;
5
6 import com.promineotech.jeeo.entity.Order;
8
9 /**
10  * @author UJJVAL
11  *
12  */
13 public class DefaultJeepOrderService implements JeepOrderService {
14
15     @Override
16     public Order createOrder(OrderRequest orderRequest) {
17         // TODO Auto-generated method stub
18         return null;
19     }
20
21 }
22
```

```
CreateOrderTest.java  *BasicJeepOrderController.java  *DefaultJeepOrderService.java ×
1  /**
4  package com.promineotech.jeeo.service;
5
6  import org.springframework.beans.factory.annotation.Autowired;
7  import com.promineotech.jeeo.dao.JeeoOrderDao;
8  import com.promineotech.jeeo.entity.Order;
9  import com.promineotech.jeeo.entity.OrderRequest;
10
11 /**
12  * @author UJJVAL
13  *
14  */
15 public class DefaultJeepOrderService implements JeeoOrderService {
16
17     @Autowired
18     private JeeoOrderDao jeepOrderDao;
19
20     @Override
21     public Order createOrder(OrderRequest orderRequest) {
22         // TODO Auto-generated method stub
23         return jeepOrderDao.createOrder(orderRequest);
24     }
25
26 }
27
```



```
CreateOrderTest.java  DefaultJeepOrderDao.java  DefaultJeepOrderService.java × JeepOrderDao.java
29
30 @Autowired
31 private JeepOrderDao jeepOrderDao;
32
33 @Transactional
34 @Override
35 public Order createOrder(OrderRequest orderRequest) {
36     Customer customer = getCustomer(orderRequest);
37     Jeep jeep = getModel(orderRequest);
38     Color color = getColor(orderRequest);
39     Engine engine = getEngine(orderRequest);
40     Tire tire = getTire(orderRequest);
41     List<Option> options = getOption(orderRequest);
42
43     BigDecimal price = jeep.getBasePrice().add(color.getPrice())
44         .add(engine.getPrice()).add(tire.getPrice());
45
46     return jeepOrderDao.saveOrder(customer, jeep, color, engine, tire, price);
47 }
48 /**
49  *
50  * @param orderRequest
51  * @return
52  */
53 private List<Option> getOption(OrderRequest orderRequest) {
54     return jeepOrderDao.fetchOptions(orderRequest.getOptions());
55 }
56
57 /**
58  *
59  * @param orderRequest
60  * @return
61  */
62 private Tire getTire(OrderRequest orderRequest) {
63     return jeepOrderDao.fetchTire(orderRequest.getTire())
64         .orElseThrow(() -> new NoSuchElementException(
65             "Tire with ID=" + orderRequest.getTire() + " was not found"));
66 }
67
```

```

46  * @param doors
47  * @return
48  */
49  Optional<Jeep> fetchModel(JeepModel model, String trim, int doors);
50
51  /**
52   * @param colorId
53   * @return
54   */
55  Optional<Color> fetchColor(String colorId);
56
57  /**
58   * @param engineId
59   * @return
60   */
61  Optional<Engine> fetchEngine(String engineId);
62
63  /**
64   * @param tireId
65   * @return
66   */
67  Optional<Tire> fetchTire(String tireId);
68
69  /**
70   * @param customer
71   * @param jeep
72   * @param color
73   * @param engine
74   * @param tire
75   * @param price
76   * @param options
77   * @return
78   */
79
80  Order saveOrder(Customer customer, Jeep jeep, Color color, Engine engine, Tire tire,
81                  BigDecimal price, List<Option> options);
82
83 }
84
```

```
CreateOrderTest.java | DefaultJeepOrderDao.java | *DefaultJeepOrderService.java | JeepOrderDao.java

31 import lombok.extern.slf4j.Slf4j;
32
33 /**
34  * @author UJJVAL
35  *
36  */
37 @Component
38 @Slf4j
39 public class DefaultJeepOrderDao implements JeepOrderDao {
40
41     @Autowired
42     private NamedParameterJdbcTemplate jdbcTemplate;
43
44     @Override
45     public Order saveOrder(Customer customer, Jeep jeep, Color color, Engine engine, Tire tire,
46         BigDecimal price, List<Option> options) {
47         // TODO Auto-generated method stub
48         SqlParams params = generateInsertSql(customer, jeep, color, engine, tire, price);
49
50         KeyHolder keyHolder = new GeneratedKeyHolder();
51         jdbcTemplate.update(params.sql, params.source, keyHolder);
52
53         Long orderPK = keyHolder.getKey().longValue();
54
55         saveOptions(options, orderPK);
56
57         /// @formatter:off
58         return Order.builder()
59             .orderPK(orderPK)
60             .customer(customer)
61             .model(jeep)
62             .color(color)
63             .engine(engine)
64             .tire(tire)
65             .options(options)
66             .price(price)
67             .build();
68     }
69 }
```

Problems | Javadoc | Declaration | Console

workspace-spring-tool-suite-4-4.16.1.RELEASE - jeep-sales/src/test/java/com/promineotech/jeep/controller/CreateOrderTest.java - Spring Tool Suite 4

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer | Run | Console | Outline

Finished after 16.43 seconds
Runs: 1/1 | Errors: 0 | Failures: 0

CreateOrderTest [Runner: JUnit 5] (2.450 s)

Failure Trace

Boot Dashboard

Type tags, projects, or working set names to match (incl. * and ? wildcards)

local

1 elements hidden by filter

```
31 @SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
32 @ActiveProfiles("test")
33 @Sql(scripts = {"classpath:flyway/migrations/V1.0_Jeep_Schema.sql",
34     "classpath:flyway/migrations/V1.1_Jeep_Data.sql"},
35     config = @SqlConfig(encoding = "utf-8"))
36 class CreateOrderTest extends CreateOrderTestSupport {
37
38     @Test
39     void testCreateOrderReturnsSuccess201() {
40         String body = createOrderBody();
41         String url = getBaseUrlForOrders();
42
43         HttpHeaders headers = new HttpHeaders();
44         headers.setContentType(MediaType.APPLICATION_JSON);
45
46         HttpEntity<String> bodyEntity = new HttpEntity<>(body, headers);
47
48         ResponseEntity<com.promineotech.jeep.entity.Order> response = getRestTemplate().exchange(url, HttpMethod.POST,
49             bodyEntity, com.promineotech.jeep.entity.Order.class);
50         assertEquals(HttpStatus.CREATED, response.getStatusCode());
51     }
52 }
```

Problems | Javadoc | Declaration | Console

<terminated> CreateOrderTest [JUnit] C:\Users\UJJVAL\Desktop\Promineo BackEnd\sts-4.16.1.RELEASE\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.j7.0.4.v20221004-1257\jre\bin\java.exe (Jan 20, 2023 11:00:15.648 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Using TestExecutionListeners: [org.springframework.test.context.junit4.SpringJUnit4ClassRunner, org.springframework.test.context.support.DependencyInjectionTestExecutionListener, org.springframework.test.context.support.DirtiesContextTestExecutionListener, org.springframework.test.context.support.SpringJUnit4ClassRunner, org.springframework.test.context.support.SpringJUnit5ClassRunner])

11:00:15.657 [main] DEBUG org.springframework.test.context.support.AbstractDirtiesContextTestExecutionListener - Before test class: context [DefaultTestContext@11:00:15.657 [main] DEBUG org.springframework.test.context.support.DependencyInjectionTestExecutionListener - Performing dependency injection for test context

Spring Boot (v2.7.6)

2023-01-20 11:00:16.575 INFO 20656 --- [main] c.p.jeep.controller.CreateOrderTest : Starting CreateOrderTest using Java 17.0.4.1 on ARNAV wi

2023-01-20 11:00:16.579 DEBUG 20656 --- [main] c.p.jeep.controller.CreateOrderTest : Running with Spring Boot v2.7.6, Spring v5.3.24

2023-01-20 11:00:16.583 INFO 20656 --- [main] c.p.jeep.controller.CreateOrderTest : The following 1 profile is active: "test"

2023-01-20 11:00:28.339 INFO 20656 --- [main] c.p.jeep.controller.CreateOrderTest : Started CreateOrderTest in 12.559 seconds (JVM running f

2023-01-20 11:00:30.391 DEBUG 20656 --- [o-auto-1-exec-1] c.p.j.c.BasicJeepOrderController : Orders= OrderRequest(customer=MORISON_LINA, model=WRANGL