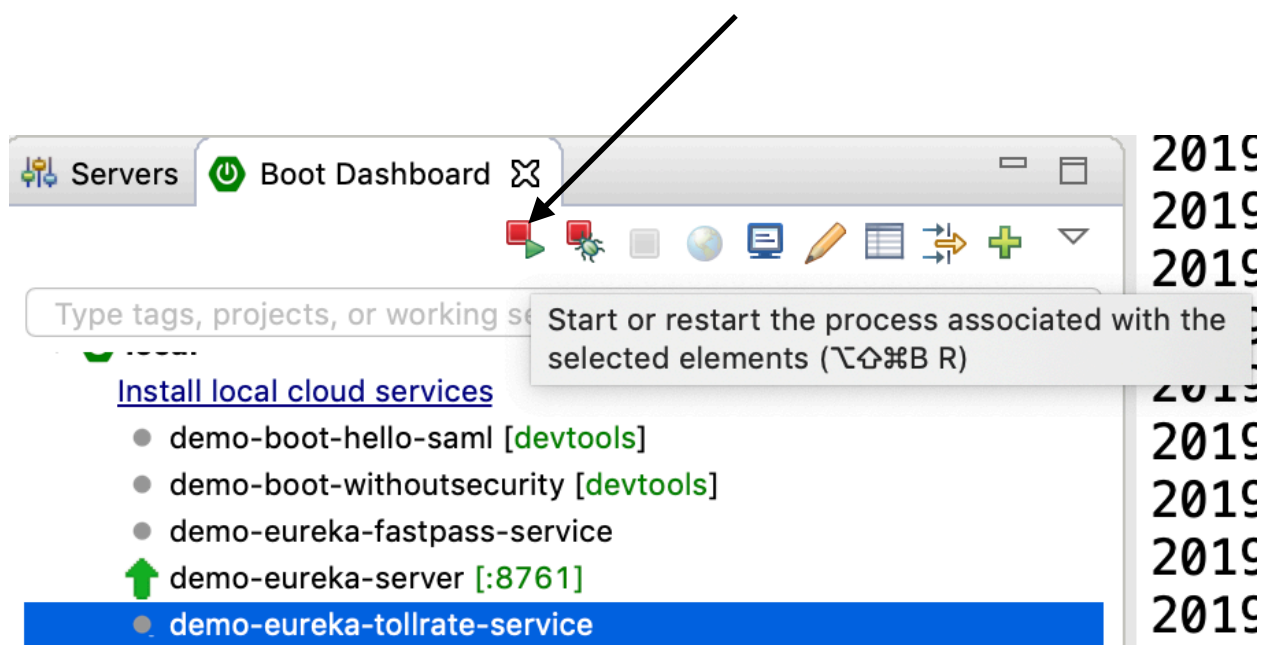


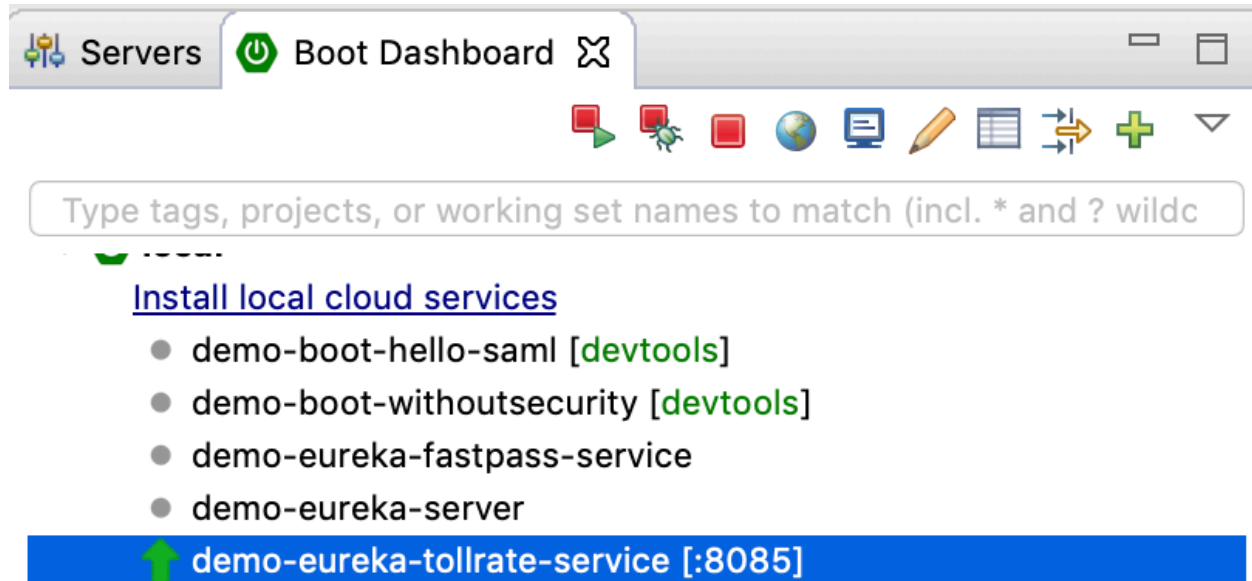
Registering Multiple Services with Eureka

Part 1 – Import & Run Tollrate Service

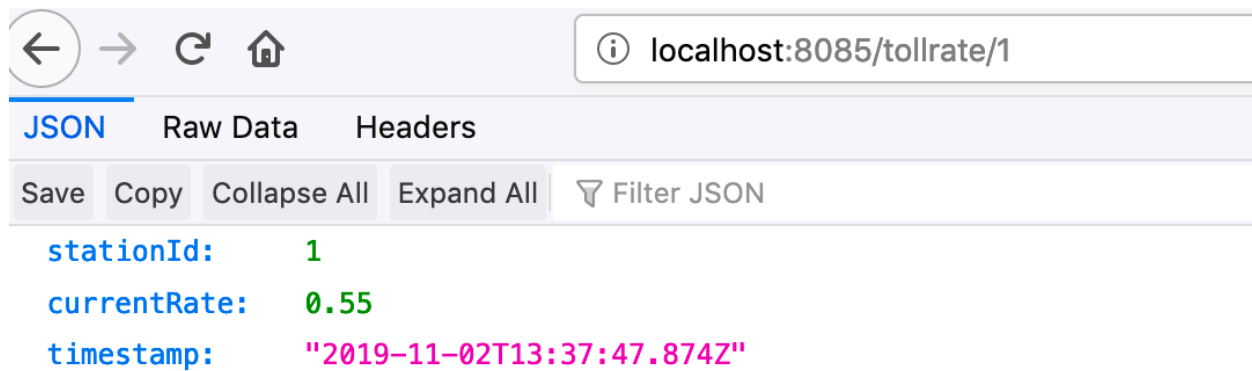
1. Import 2 projects **demo-eureka-tollrate-service** as Maven project
2. Start **demo-eureka-tollrate-service** application



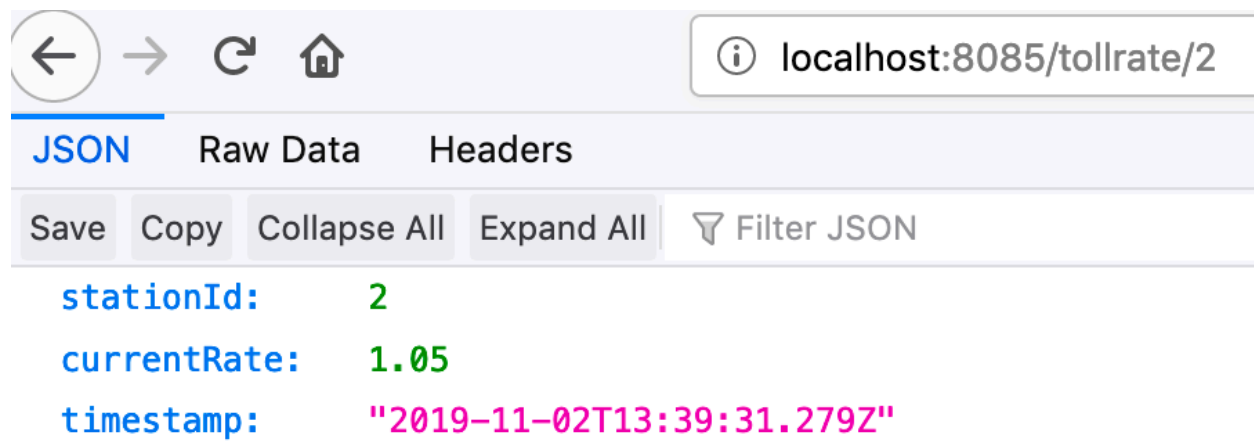
3. It starts on port# 8085



4. Visit <http://localhost:8085/tollrate/1>



5. Change param values to 2, 3 and observe the appropriate toll rate



The screenshot shows a web browser window with a REST client interface. The address bar displays `localhost:8085/tollrate/2`. Below the address bar, there are tabs for `JSON`, `Raw Data`, and `Headers`. The `JSON` tab is selected. Below the tabs, there are buttons for `Save`, `Copy`, `Collapse All`, `Expand All`, and a `Filter JSON` button. The JSON response is displayed in a preformatted style:

```
{
  "stationId": 2,
  "currentRate": 1.05,
  "timestamp": "2019-11-02T13:39:31.279Z"
}
```

Part 2 – Register Tollrate Service with Eureka

6. Add spring-cloud-eureka dependency

pom.xml

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-eureka</artifactId>
  <version>1.4.5.RELEASE</version>
</dependency>
```

7. Print the Station ID in the Controller

TollRateController.java

Line #16

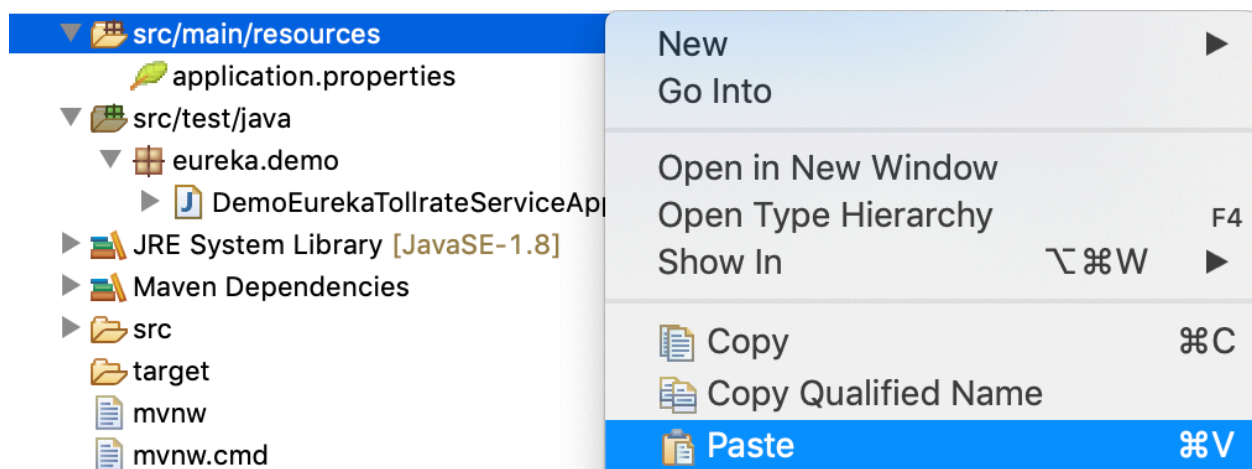
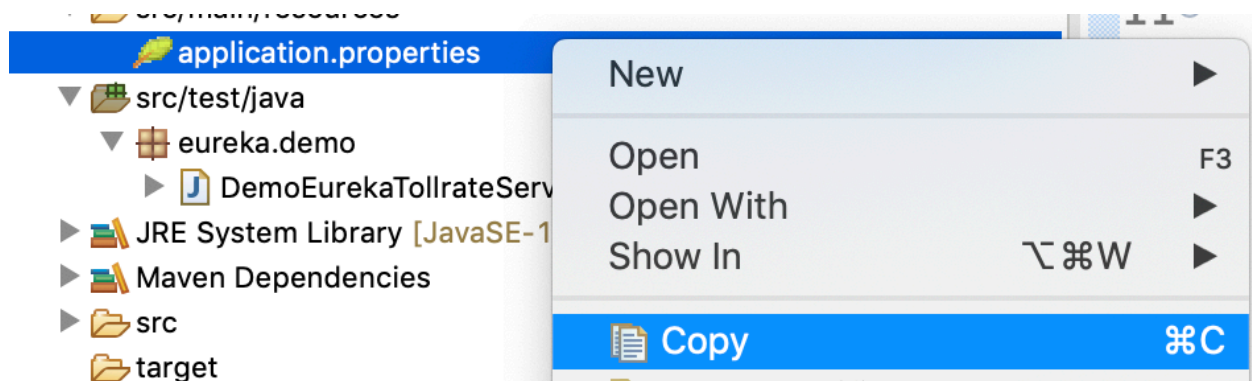
```
10 @RestController
11 public class TollRateController {
12
13     @RequestMapping("/tollrate/{stationId}")
14     public TollRate GetTollRate(@PathVariable int stationId) {
15
16         System.out.println("Station requested: " + stationId);
17     }
```

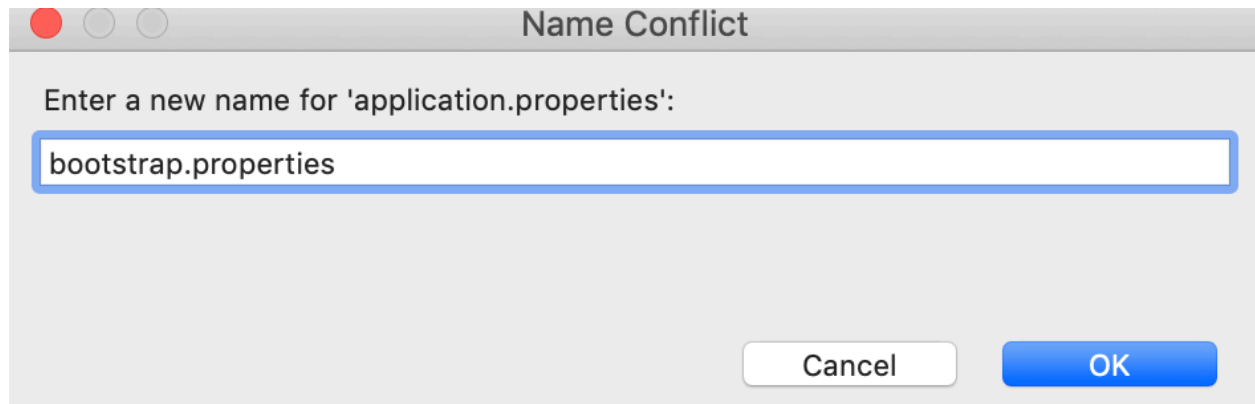
8. Enable Eureka Client for this Micro service to register with Eureka Server using Annotation

DemoEurekaTollrateServiceApplication.java

```
@EnableEurekaClient
@SpringBootApplication
public class DemoEurekaTollrateServiceApplication {
```

9. Create bootstrap.properties





10. App Name in

bootstrap.properties

```
spring.application.name = demo-toll-service
```

11. Add these properties and change port number

To be able to start multiple instances, port# can be random for every instance

application.properties

```
# eureka.client.serviceUrl.defaultZone = http://localhost:8761/eureka

eureka.client.register-with-eureka = true
eureka.client.fetch-registry = true
eureka.client.healthcheck.enabled = true

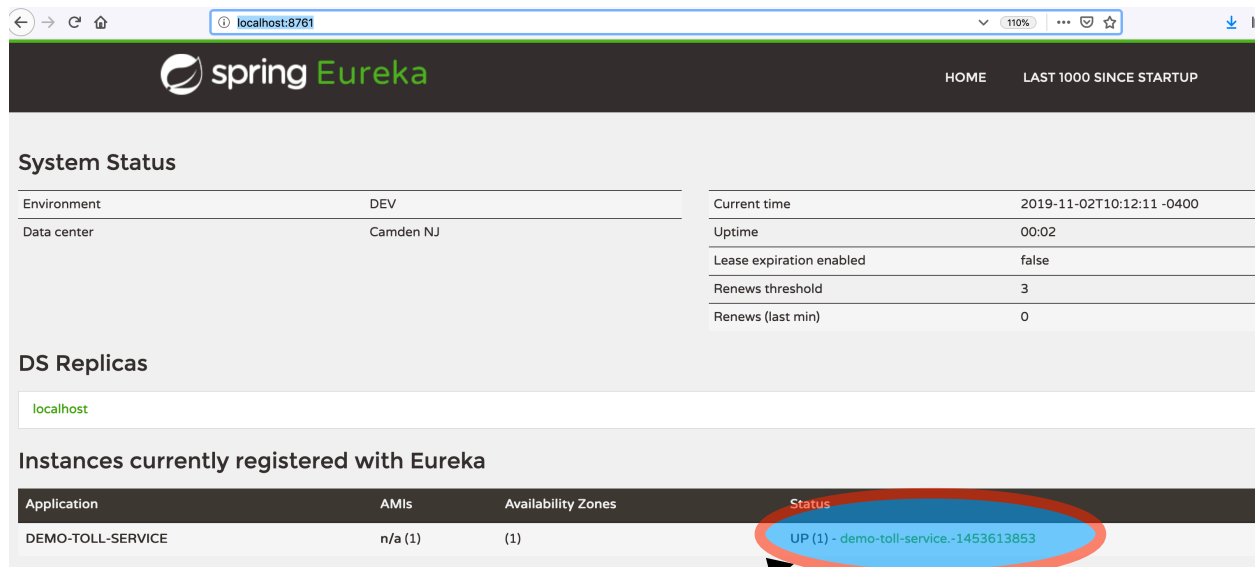
eureka.instance.instance-id = ${spring.application.name}.${random.int}

server.port = 0
```

12. RUN the APP

13. Visit ***http://localhost:8761/***

Observe that tollrate service is registered in the Eureka Dashboard



The screenshot shows the Spring Eureka Dashboard at `localhost:8761`. The dashboard has a dark header with the 'spring Eureka' logo and navigation links for 'HOME' and 'LAST 1000 SINCE STARTUP'. Below the header, there are two main sections: 'System Status' and 'Instances currently registered with Eureka'.

System Status

| | | | |
|-------------|-----------|--------------------------|---------------------------|
| Environment | DEV | Current time | 2019-11-02T10:12:11 -0400 |
| Data center | Camden NJ | Uptime | 00:02 |
| | | Lease expiration enabled | false |
| | | Renews threshold | 3 |
| | | Renews (last min) | 0 |

DS Replicas

localhost

Instances currently registered with Eureka

| Application | AMIs | Availability Zones | Status |
|-------------------|---------|--------------------|--|
| DEMO-TOLL-SERVICE | n/a (1) | (1) | UP (1) - demo-toll-service.-1453613853 |

An arrow points from the text 'Hover your mouse on' to the 'Status' column header, and another arrow points from the text 'and observe the URL' to the status value 'UP (1) - demo-toll-service.-1453613853'.

Hover your mouse on and observe the URL

14. Change hostname

application.properties

```
eureka.instance.hostname = localhost
```

15. RUN the APP

16. Refresh Eureka Dashboard and observe URL for the Toll Service as above

