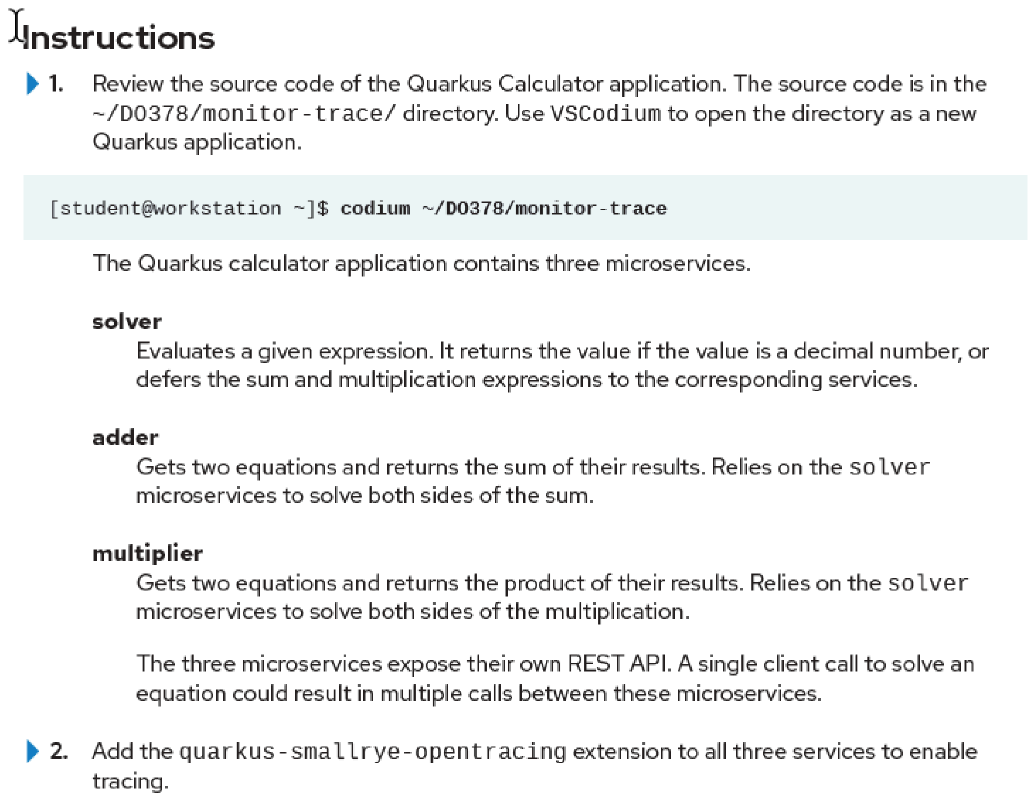
LAB 23: QUARKUS MONITOR TRACE

Autor: José Díaz

Github Repo: https://github.com/joedayz/quarkus-bcp-2025.git

Abre el proyecto **monitor-trace**



Instructions

1. Review the source code of the Quarkus Calculator application. The source code is in the `~/DO378/monitor-trace/` directory. Use `VSCodium` to open the directory as a new Quarkus application.

```
[student@workstation ~]$ codium ~/DO378/monitor-trace
```

The Quarkus calculator application contains three microservices.

solver
Evaluates a given expression. It returns the value if the value is a decimal number, or defers the sum and multiplication expressions to the corresponding services.

adder
Gets two equations and returns the sum of their results. Relies on the `solver` microservices to solve both sides of the sum.

multiplier
Gets two equations and returns the product of their results. Relies on the `solver` microservices to solve both sides of the multiplication.

The three microservices expose their own REST API. A single client call to solve an equation could result in multiple calls between these microservices.

2. Add the `quarkus-smallrye-opentracing` extension to all three services to enable tracing.

2.1. Inspect the `~/DO378/monitor-trace/add-tracing.sh` script. This script adds the `quarkus-smallrye-opentracing` Quarkus extension to each of the microservices.

```
...output omitted...
echo "Adding tracing extension to the 'solver' project "
cd solver
mvn quarkus:add-extension -Dextension=smallrye-opentracing
...output omitted...
```

2.2. Run the `~/DO378/monitor-trace/add-tracing.sh` script.

```
[student@workstation ~]$ sh ~/DO378/monitor-trace/add-tracing.sh
Adding tracing extension to the 'solver' project
...output omitted...
Extension io.quarkus:quarkus-smallrye-opentracing has been installed
...output omitted...
[INFO] BUILD SUCCESS
...output omitted...
```

3. Start a local instance of Jaeger by using **podman**.

3.1. Inspect the `~/DO378/monitor-trace/jaeger.sh` script, which starts Jaeger in a container.

3.2. In a new terminal on the `workstation` VM, start the Jaeger container by using the `jaeger.sh` script.

```
[student@workstation ~]$ sh ~/DO378/monitor-trace/jaeger.sh
Starting the all-in-one Jaeger container
...output omitted...
"Starting jaeger-collector HTTP server","http host-port":":14268"}
...output omitted...
"Query server started","port":16686,"addr":":16686"}
"Health Check state change","status":"ready"}
...output omitted...
```

4. Configure the `adder` service to send tracing information to Jaeger.

4.1. Edit the `~/DO378/monitor-trace/adder/src/main/resources/application.properties` file, and add the following properties.

```
quarkus.jaeger.service-name=adder
quarkus.jaeger.sampler-type=const
quarkus.jaeger.sampler-param=1
quarkus.log.console.format=%d{HH:mm:ss} %-5p traceId=%X{traceId}, spanId=
%X{spanId}, sampled=%X{sampled} [%c{2.}] (%t) %s%e%n
quarkus.jaeger.endpoint=http://localhost:14268/api/traces
quarkus.jaeger.propagation=b3
quarkus.jaeger.reporter-log-spans=true
```

The `quarkus.jaeger.endpoint` property configures the URL of the Jaeger collector, which gathers tracing data from the microservices.

Important

Verify that the preceding properties do not contain trailing spaces. Otherwise, the application might not be able to send traces to Jaeger.

4.2. Verify that the same configuration exists in the `multiplier` and `solver` projects.

Important

The value of the `quarkus.jaeger.service-name` property must be unique. This value identifies and labels the traces in the Jaeger web console.

5. Start the three microservices.

Inspect and run the `~/DO378/monitor-trace/start.sh` script in a new terminal on the `workstation` VM.

```
[student@workstation ~]$ sh ~/DO378/monitor-trace/start.sh
Starting the 'solver' project
...output omitted...
Starting the 'adder' project
...output omitted...
Starting the 'multiplier' project
...output omitted...
Press enter to Terminate
...output omitted...
```

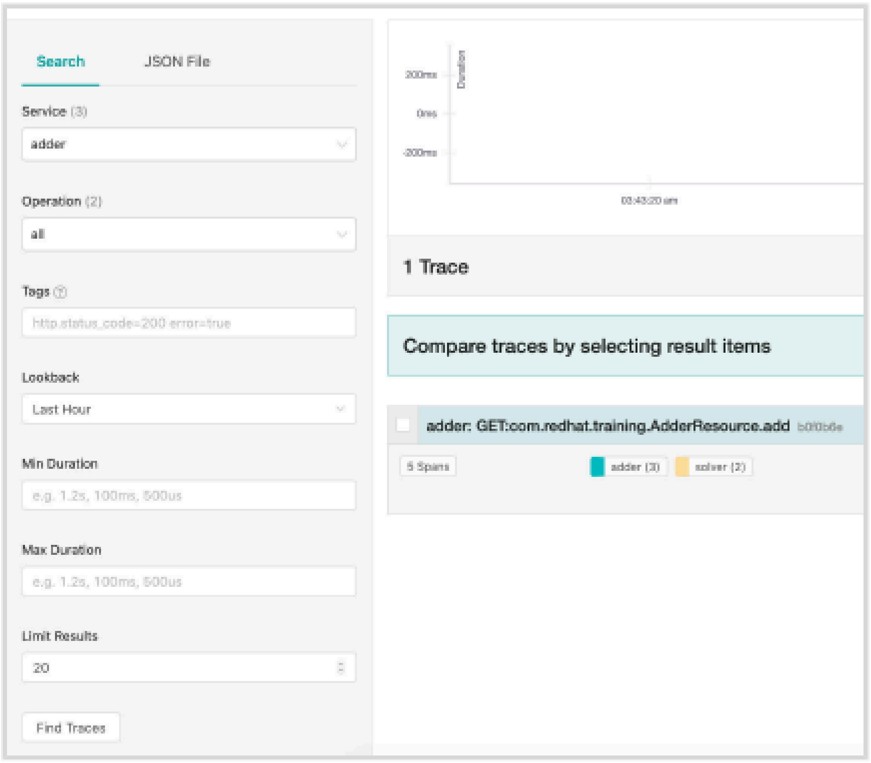
6. Capture traces by invoking the REST endpoints exposed by the microservices. Use the Jaeger web console to visualize the traces and execution timing.

6.1. On the `workstation` VM, navigate to the Jaeger web console at `http://localhost:16686` in a web browser. You should not see any traces because you have not invoked any endpoint in the application.

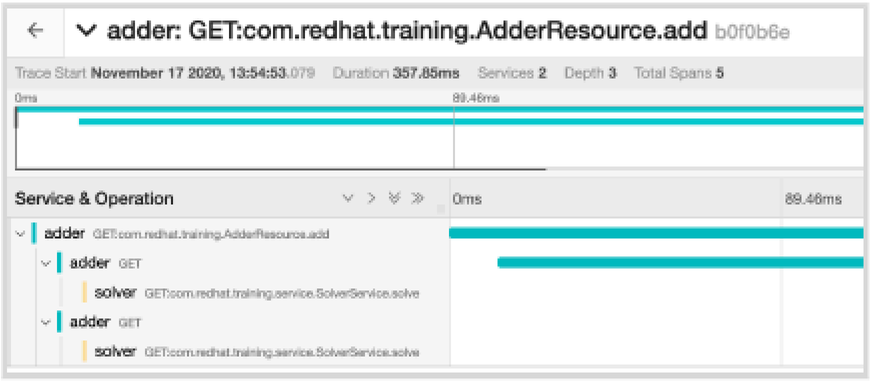
6.2. Open a new terminal on the `workstation` VM, and invoke the endpoint for the `adder` microservice.

```
[student@workstation ~]$ curl "http://localhost:8081/adder/5/3"; echo
8.0
```

6.3. Refresh the Jaeger web console. Select the `adder` service from the Service field in the Search panel on the left. Click Find Traces to view the trace.



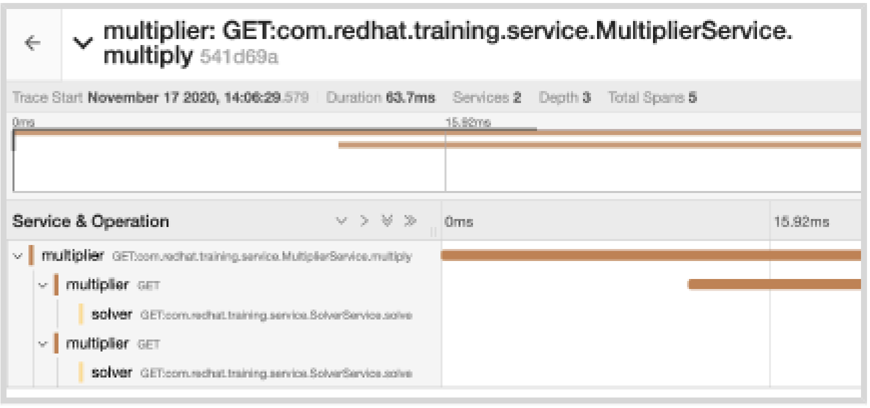
6.4. Click the adder:GET:com.redhat.training.AdderResource.add trace to view the details of the trace.


adder: GET:com.redhat.training.AdderResource.add b0f0b6e
Trace Start November 17 2020, 13:54:53.079 Duration 357.85ms Services 2 Depth 3 Total Spans 5
0ms
89.46ms
Service & Operation
adder GET:com.redhat.training.AdderResource.add
adder GET
solver GET:com.redhat.training.service.SolverService.solve
adder GET
solver GET:com.redhat.training.service.SolverService.solve

6.5. Switch to the terminal where you ran the `curl` command and invoke the endpoint for the `multiplier` microservice.

```
[student@workstation ~]$ curl "http://localhost:8082/multiplier/5/4"; echo
20.0
```

6.6. Find the trace for this invocation by selecting the `multiplier` service from the Service field in the Jaeger home page, and click Find Traces. Click the multiplier:GET:com.redhat.training.MultiplierService.multiply trace to view the details of the trace. The output should be as follows.



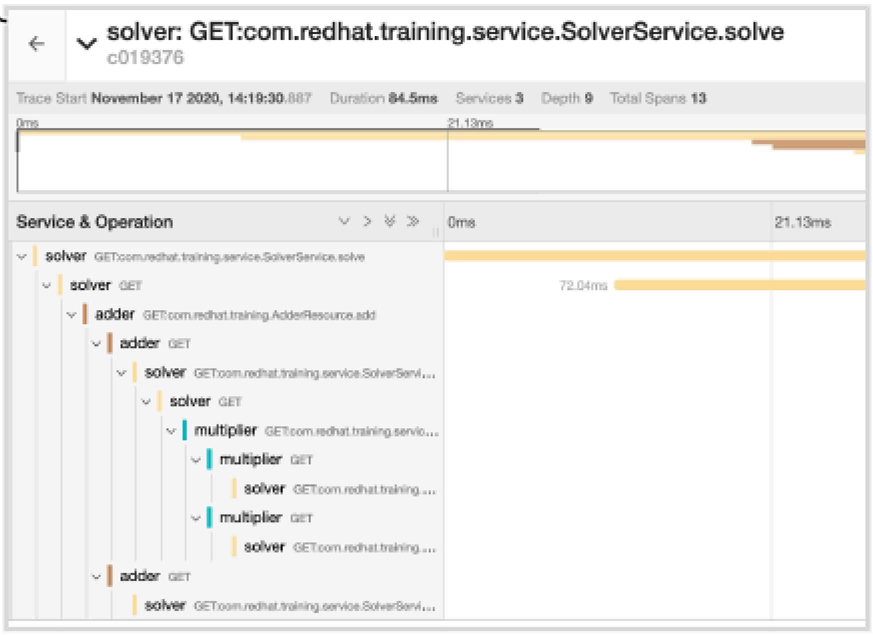
```
multiplier: GET:com.redhat.training.service.MultiplierService.multiply 541d69a
Trace Start November 17 2020, 14:06:29.579   Duration 63.7ms   Services 2   Depth 3   Total Spans 5

Service & Operation                                                    0ms          15.92ms
multiplier  GET:com.redhat.training.service.MultiplierService.multiply
  multiplier  GET
    solver  GET:com.redhat.training.service.SolverService.solve
  multiplier  GET
    solver  GET:com.redhat.training.service.SolverService.solve
```

6.7. Invoke the endpoint for the `solver` microservice. This service can take compound equations with addition and multiplication terms as input.

```
[student@workstation ~]$ curl "http://localhost:8080/solver/5*4+3"; echo
23.0
```

6.8. Find the trace for this invocation by selecting the `solver` service from the Service field in the Jaeger home page, and click Find Traces. Click the solver:GET:com.redhat.training.SolverService.solve trace to view the details of the trace. The output should be as follow.



solver: GET:com.redhat.training.service.SolverService.solve
c019376

Trace Start **November 17 2020, 14:19:30**.887 Duration **84.5ms** Services **3** Depth **9** Total Spans **13**

```
Service & Operation                                              0ms              21.13ms
solver GET:com.redhat.training.service.SolverService.solve
  solver GET                                                          72.04ms
    adder GET:com.redhat.training.AdderResource.add
      adder GET
        solver GET:com.redhat.training.service.SolverServi...
          solver GET
            multiplier GET:com.redhat.training.servic...
              multiplier GET
                solver GET:com.redhat.training....
              multiplier GET
                solver GET:com.redhat.training....
      adder GET
        solver GET:com.redhat.training.service.SolverServi...
```

7. Clean up. Stop the microservices and the Jaeger container.

7.1. To stop the three microservices, press `Enter` in the terminal window where you ran the `start.sh` script.

7.2. To stop the Jaeger container, press `Ctrl+C` in the terminal window where you ran the `jaeger.sh` script.

Finish

On the `workstation` machine, use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish monitor-trace
```

This concludes the section.

enjoy!

Jose