Modern Cloud-native Java runtimes performance monitoring on Red Hat Openshift

Developing the Quarkus Application

In this section you will:

- Develop a REST API with Quarkus that consumes memory and CPU
- Add a Statistics persistent entity to store metrics in a PostgreSQL database
- Configure the application
- Develop some tests to validate the behavior of the application
- Test and run the application locally
- Check a few metrics locally

You should have a directory called `quarkus-app` inside your project repo (`$PROJECT_SOURCE/`). This is the root of the Quarkus microservice source code that we will be working on during this this section.

The Quarkus REST Resource

The Quarkus application is made of a simple REST resource that consumes memory and CPU. The resource is defined in the `QuarkusResource` class.

Create a new file called `QuarkusResource.java` , under the `src/main/java/io/containerapps/javaruntime/workshop/quarkus` directory.

> Maven based projects follows a standard directory layout (https://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html).
>
> Application classes containing domain and logic **must** reside under the `src/main/java/` directory.
>
> ```
> .
> ├── pom.xml
> ├── src
> │   ├── main
> │   │   ├── java
> │   │   │   └── ...
> ```

Then add the following to the header of this class file (replacing any existing content auto-generated by the IDE when you first created this class file).

As you can see, it’s a JAX-RS resource that exposes the `/quarkus` path.

Header of the Quarkus REST Resource

```java
package io.containerapps.javaruntime.workshop.quarkus;

import javax.ws.rs.DefaultValue;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;
import java.lang.System.Logger;
import java.time.Duration;
import java.time.Instant;
import java.util.HashMap;
import java.util.List;

import static java.lang.System.Logger.Level.INFO;
import static java.lang.invoke.MethodHandles.lookup;

@Path("/quarkus")
@Produces(MediaType.TEXT_PLAIN)
public class QuarkusResource {

    private static final Logger LOGGER = System.getLogger(lookup().lookupClass().getName());

    private final StatisticsRepository repository;

    public QuarkusResource(StatisticsRepository statisticsRepository) {
        this.repository = statisticsRepository;
    }

    //  ====> insert the remaining code blocks under his line <====
```