Lesson 03 Demo 01

Adding Instrumentation to a Java Application

Objective: To add instrumentation to a Java application for enabling metric collection and visualizing the metrics using Prometheus

Tools required: Linux operating system, Git, and Maven

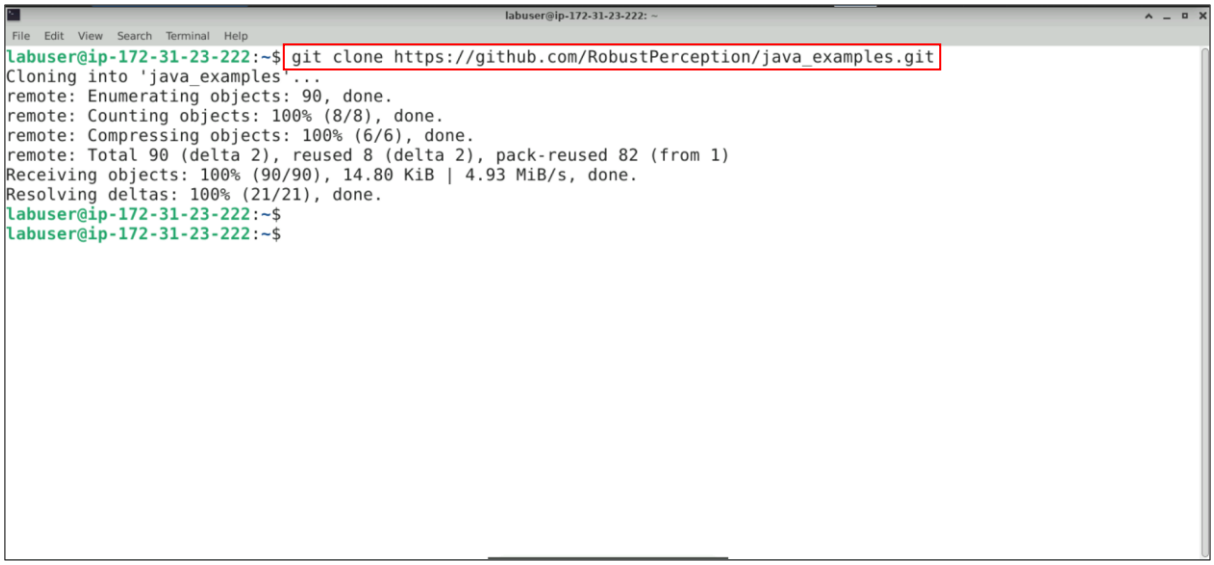
Prerequisites: Refer to Demo 02 of Lesson 01 for setting up a Prometheus server

Steps to be followed:

1. Clone the GitHub repository for the Java application
2. Build the Java application using Maven
3. Run the Java application to expose Prometheus metrics
4. Visualize the metrics using the Prometheus UI

Step 1: Clone the GitHub repository for the Java application

1.1 Navigate to the terminal in the system and run the following command to clone the repository from the given URL to the local machine:
git clone https://github.com/RobustPerception/java_examples.git



Note: Ensure that **Git** is installed. If not, run the following command to install it:
sudo apt-get install git

1.2 Change the directory to the **java_simple** folder within the **java_examples** project using the following command:
cd java_examples/java_simple/

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
labuser@ip-172-31-23-222:~$ git clone https://github.com/RobustPerception/java_examples.git
Cloning into 'java_examples'...
remote: Enumerating objects: 90, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 90 (delta 2), reused 8 (delta 2), pack-reused 82 (from 1)
Receiving objects: 100% (90/90), 14.80 KiB | 4.93 MiB/s, done.
Resolving deltas: 100% (21/21), done.
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$ cd java_examples/java_simple/
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
```

Step 2: Build the Java application using Maven

2.1 Build the Java application using Maven to create a JAR file using the following command:
mvn package

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
labuser@ip-172-31-23-222:~$ cd java_examples/java_simple/
labuser@ip-172-31-23-222:~/java_examples/java_simple$ ls
README.md  pom.xml  src
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----------< io.robustperception.java_examples:java_simple >------------
[INFO] Building java_simple 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-
2.6.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2
.6.pom (8.1 kB at 11 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/23/maven-plugins-23.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/23/maven-plugins-23.pom (9.2 kB at 20
9 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/22/maven-parent-22.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/22/maven-parent-22.pom (30 kB at 402 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/11/apache-11.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/11/apache-11.pom (15 kB at 322 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-
2.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2
.6.jar (30 kB at 278 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.1/maven-compiler-plugin-3.
1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.1/maven-compiler-plugin-3.1
.pom (10 kB at 243 kB/s)
```

The application is successfully built as shown below:

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] META-INF/MANIFEST.MF already added, skipping
[INFO] META-INF/ already added, skipping
[INFO] org/ already added, skipping
[INFO] org/eclipse/ already added, skipping
[INFO] org/eclipse/jetty/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.eclipse.jetty/ already added, skipping
[INFO] about.html already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] META-INF/MANIFEST.MF already added, skipping
[INFO] META-INF/ already added, skipping
[INFO] org/ already added, skipping
[INFO] org/eclipse/ already added, skipping
[INFO] org/eclipse/jetty/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.eclipse.jetty/ already added, skipping
[INFO] about.html already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  25.156 s
[INFO] Finished at: 2024-09-06T10:43:58Z
[INFO] ------------------------------------------------------------------------
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
```

Note: Ensure that the **mvn** tool is already installed on the system

2.2 Run the Java application using the following command:
nohup java -jar target/java_simple-1.0-SNAPSHOT-jar-with-dependencies.jar > /dev/null 2>&1 &

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] META-INF/MANIFEST.MF already added, skipping
[INFO] META-INF/ already added, skipping
[INFO] org/ already added, skipping
[INFO] org/eclipse/ already added, skipping
[INFO] org/eclipse/jetty/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.eclipse.jetty/ already added, skipping
[INFO] about.html already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  25.156 s
[INFO] Finished at: 2024-09-06T10:43:58Z
[INFO] ------------------------------------------------------------------------
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ nohup java -jar target/java_simple-1.0-SNAPSHOT-jar-with-dependencies.jar > /dev/nul
l 2>&1 &
[1] 3710
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ curl localhost:1234/
Hello World!
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
```

2.3 Validate that the Java application is running in the background by printing **Hello World!** using the following command:
curl localhost:1234/

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] META-INF/MANIFEST.MF already added, skipping
[INFO] META-INF/ already added, skipping
[INFO] org/ already added, skipping
[INFO] org/eclipse/ already added, skipping
[INFO] org/eclipse/jetty/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.eclipse.jetty/ already added, skipping
[INFO] about.html already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE.txt already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  25.156 s
[INFO] Finished at: 2024-09-06T10:43:58Z
[INFO] ------------------------------------------------------------------------
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ nohup java -jar target/java_simple-1.0-SNAPSHOT-jar-with-dependencies.jar > /dev/nul
l 2>&1 &
[1] 3710
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ curl localhost:1234/
Hello World!
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
```

2.4 Run the following command to verify the metrics exposed by the Java application:

curl localhost:1234/metrics

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ curl localhost:1234/
Hello World!
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ curl localhost:1234/metrics
# HELP jvm_memory_pool_allocated_bytes_total Total bytes allocated in a given JVM memory pool. Only updated after GC, not continuously.
# TYPE jvm_memory_pool_allocated_bytes_total counter
# HELP jvm_classes_loaded The number of classes that are currently loaded in the JVM
# TYPE jvm_classes_loaded gauge
jvm_classes_loaded 2082.0
# HELP jvm_classes_loaded_total The total number of classes that have been loaded since the JVM has started execution
# TYPE jvm_classes_loaded_total counter
jvm_classes_loaded_total 2082.0
# HELP jvm_classes_unloaded_total The total number of classes that have been unloaded since the JVM has started execution
# TYPE jvm_classes_unloaded_total counter
jvm_classes_unloaded_total 0.0
# HELP jvm_threads_current Current thread count of a JVM
# TYPE jvm_threads_current gauge
jvm_threads_current 15.0
# HELP jvm_threads_daemon Daemon thread count of a JVM
# TYPE jvm_threads_daemon gauge
jvm_threads_daemon 5.0
# HELP jvm_threads_peak Peak thread count of a JVM
# TYPE jvm_threads_peak gauge
jvm_threads_peak 15.0
# HELP jvm_threads_started_total Started thread count of a JVM
# TYPE jvm_threads_started_total counter
jvm_threads_started_total 15.0
# HELP jvm_threads_deadlocked Cycles of JVM-threads that are in deadlock waiting to acquire object monitors or ownable synchronizers
```

```
labuser@ip-172-31-23-222: ~/java_examples/java_simple
File  Edit  View  Search  Terminal  Help
jvm_buffer_pool_used_bytes{pool="mapped",} 0.0
jvm_buffer_pool_used_bytes{pool="direct",} 57344.0
jvm_buffer_pool_used_bytes{pool="mapped - 'non-volatile memory'",} 0.0
# HELP jvm_buffer_pool_capacity_bytes Bytes capacity of a given JVM buffer pool.
# TYPE jvm_buffer_pool_capacity_bytes gauge
jvm_buffer_pool_capacity_bytes{pool="mapped",} 0.0
jvm_buffer_pool_capacity_bytes{pool="direct",} 57344.0
jvm_buffer_pool_capacity_bytes{pool="mapped - 'non-volatile memory'",} 0.0
# HELP jvm_buffer_pool_used_buffers Used buffers of a given JVM buffer pool.
# TYPE jvm_buffer_pool_used_buffers gauge
jvm_buffer_pool_used_buffers{pool="mapped",} 0.0
jvm_buffer_pool_used_buffers{pool="direct",} 4.0
jvm_buffer_pool_used_buffers{pool="mapped - 'non-volatile memory'",} 0.0
# HELP jvm_info JVM version info
# TYPE jvm_info gauge
jvm_info{version="21.0.3+9-Ubuntu-1ubuntu122.04.1",vendor="Ubuntu",runtime="OpenJDK Runtime Environment",} 1.0
# HELP hello_worlds_total Number of hello worlds served.
# TYPE hello_worlds_total counter
hello_worlds_total 1.0
# HELP jvm_gc_collection_seconds Time spent in a given JVM garbage collector in seconds.
# TYPE jvm_gc_collection_seconds summary
jvm_gc_collection_seconds_count{gc="G1 Young Generation",} 1.0
jvm_gc_collection_seconds_sum{gc="G1 Young Generation",} 0.016
jvm_gc_collection_seconds_count{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_count{gc="G1 Old Generation",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Old Generation",} 0.0
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$
```

Step 3: Run the Java application to expose Prometheus metrics

3.1 Use the following commands to change the directory from **java_examples** to **prometheus**. Then, open the **java-metrics.yaml** file for editing using **Vim** or any preferred editor:

cd prometheus

sudo vim java-metrics.yaml

```
labuser@ip-172-31-23-222: ~/prometheus
File  Edit  View  Search  Terminal  Help
jvm_buffer_pool_used_buffers{pool="direct",} 4.0
jvm_buffer_pool_used_buffers{pool="mapped - 'non-volatile memory'",} 0.0
# HELP jvm_info JVM version info
# TYPE jvm_info gauge
jvm_info{version="21.0.3+9-Ubuntu-1ubuntu122.04.1",vendor="Ubuntu",runtime="OpenJDK Runtime Environment",} 1.0
# HELP hello_worlds_total Number of hello worlds served.
# TYPE hello_worlds_total counter
hello_worlds_total 1.0
# HELP jvm_gc_collection_seconds Time spent in a given JVM garbage collector in seconds.
# TYPE jvm_gc_collection_seconds summary
jvm_gc_collection_seconds_count{gc="G1 Young Generation",} 1.0
jvm_gc_collection_seconds_sum{gc="G1 Young Generation",} 0.016
jvm_gc_collection_seconds_count{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_count{gc="G1 Old Generation",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Old Generation",} 0.0
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ cd ..
labuser@ip-172-31-23-222:~/java_examples$ cd ..
labuser@ip-172-31-23-222:~$ ls
Desktop    Pictures   alertmanager-0.27.0.linux-arm64  myDir                          prometheus.yml
Documents  Public     elk.sh                           prometheus                     pushgateway-1.9.0.linux-arm64
Downloads  Templates  java_examples                    prometheus-2.54.0.linux-arm64  snap
Music      Videos     kubectl.sha256                   prometheus-docker              thinclient_drives
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$ cd prometheus
labuser@ip-172-31-23-222:~/prometheus$
labuser@ip-172-31-23-222:~/prometheus$ sudo vim java-metrics.yaml
```

3.2 Press **I** to enter **INSERT** mode. Then, copy and paste the following configuration into the file:

scrape_configs:
- job_name: 'my-java-app'
metrics_path: '/metrics'
static_configs:
- targets: ['localhost:1234']

```
labuser@ip-172-31-23-222: ~/prometheus
File  Edit  View  Search  Terminal  Help
scrape_configs:
  - job_name: 'my-java-app'
    metrics_path: '/metrics'
    static_configs:
      - targets: ['localhost:1234']
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                    6,1           All
```

Note: Press **esc** and type **:wq** to save and exit the file

3.3 Start Prometheus in the background with the configuration from **java-metrics.yaml** using the following command:
sudo nohup ./prometheus --config.file=java-metrics.yaml > /dev/null 2>&1 &

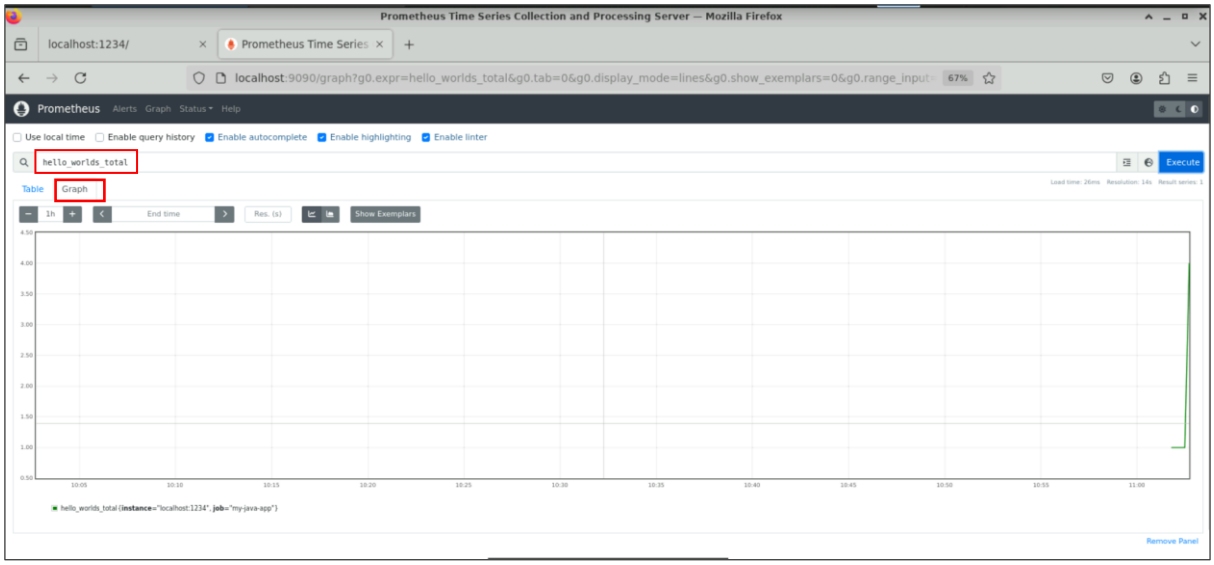
```
labuser@ip-172-31-23-222: ~/prometheus
File  Edit  View  Search  Terminal  Help
# HELP hello_worlds_total Number of hello worlds served.
# TYPE hello_worlds_total counter
hello_worlds_total 1.0
# HELP jvm_gc_collection_seconds Time spent in a given JVM garbage collector in seconds.
# TYPE jvm_gc_collection_seconds summary
jvm_gc_collection_seconds_count{gc="G1 Young Generation",} 1.0
jvm_gc_collection_seconds_sum{gc="G1 Young Generation",} 0.016
jvm_gc_collection_seconds_count{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Concurrent GC",} 0.0
jvm_gc_collection_seconds_count{gc="G1 Old Generation",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Old Generation",} 0.0
labuser@ip-172-31-23-222:~/java_examples/java_simple$
labuser@ip-172-31-23-222:~/java_examples/java_simple$ cd ..
labuser@ip-172-31-23-222:~/java_examples$ cd ..
labuser@ip-172-31-23-222:~$ ls
Desktop    Pictures   alertmanager-0.27.0.linux-arm64  myDir                          prometheus.yml
Documents  Public     elk.sh                           prometheus                     pushgateway-1.9.0.linux-arm64
Downloads  Templates  java_examples                    prometheus-2.54.0.linux-arm64  snap
Music      Videos     kubectl.sha256                   prometheus-docker              thinclient_drives
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$
labuser@ip-172-31-23-222:~$ cd prometheus
labuser@ip-172-31-23-222:~/prometheus$
labuser@ip-172-31-23-222:~/prometheus$ sudo vim java-metrics.yaml
labuser@ip-172-31-23-222:~/prometheus$
labuser@ip-172-31-23-222:~/prometheus$
labuser@ip-172-31-23-222:~/prometheus$ sudo nohup ./prometheus --config.file=java-metrics.yaml > /dev/null 2>&1 &
[2] 5895
labuser@ip-172-31-23-222:~/prometheus$
```

Note: You can also run the command **sudo ./prometheus --config.file=java-metric.yaml**

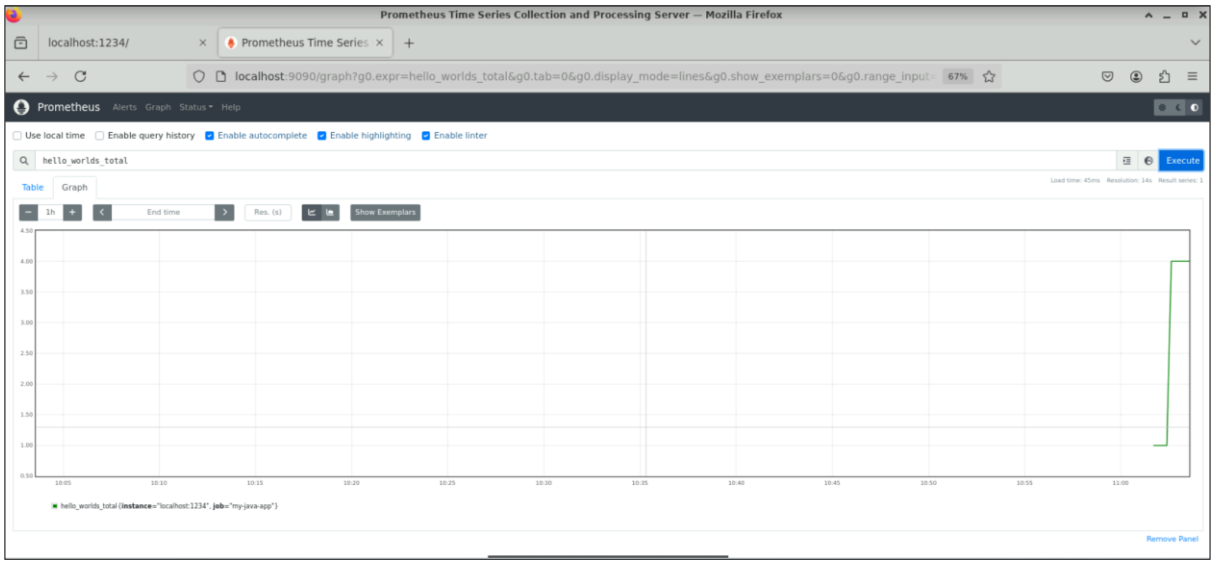
Step 4: Visualize the metrics using the Prometheus UI

4.1 Navigate to the preferred browser and enter the URL **localhost:1234** to display **Hello World** as shown below:

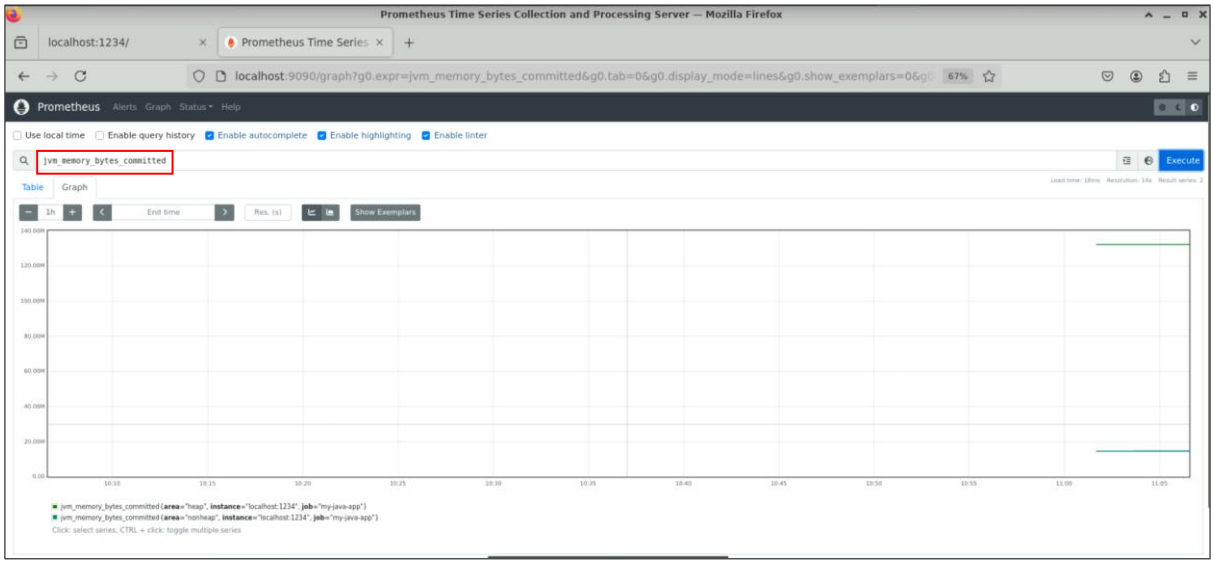
4.2 Open a new tab, enter the link **http://localhost:9090**, execute **hello_worlds_total**, and click on the **Graph** tab



4.3 Access the application on **http://localhost:1234/** multiple times to observe the increment in counter metrics



4.4 Execute the following metrics in the expression browser, then enter the link **localhost:1234** multiple times to observe the increment in the graph as shown: **jvm_memory_bytes_committed**



By following these steps, you have successfully instrumented a Java application to expose Prometheus metrics, set up a Prometheus server to scrape the metrics, and visualized them using the Prometheus UI.