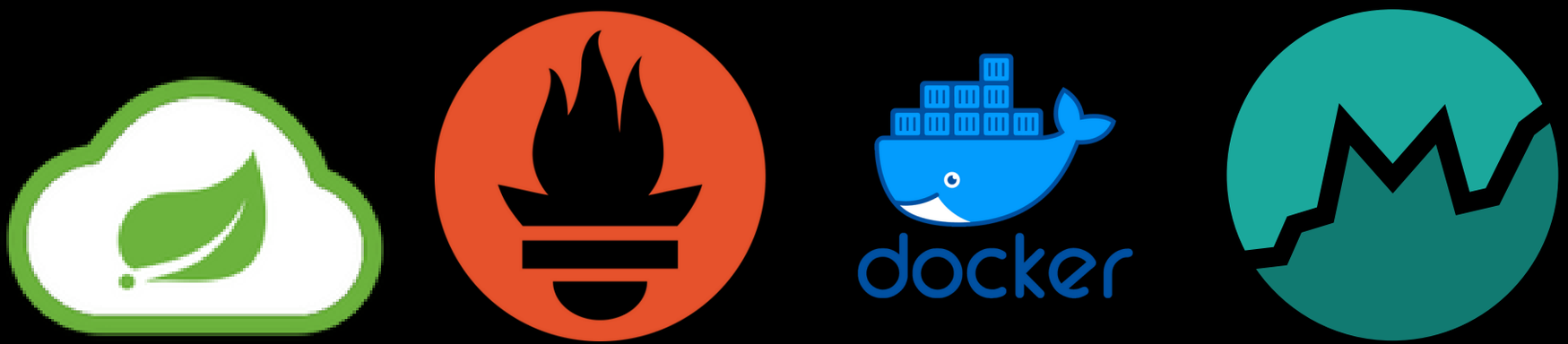
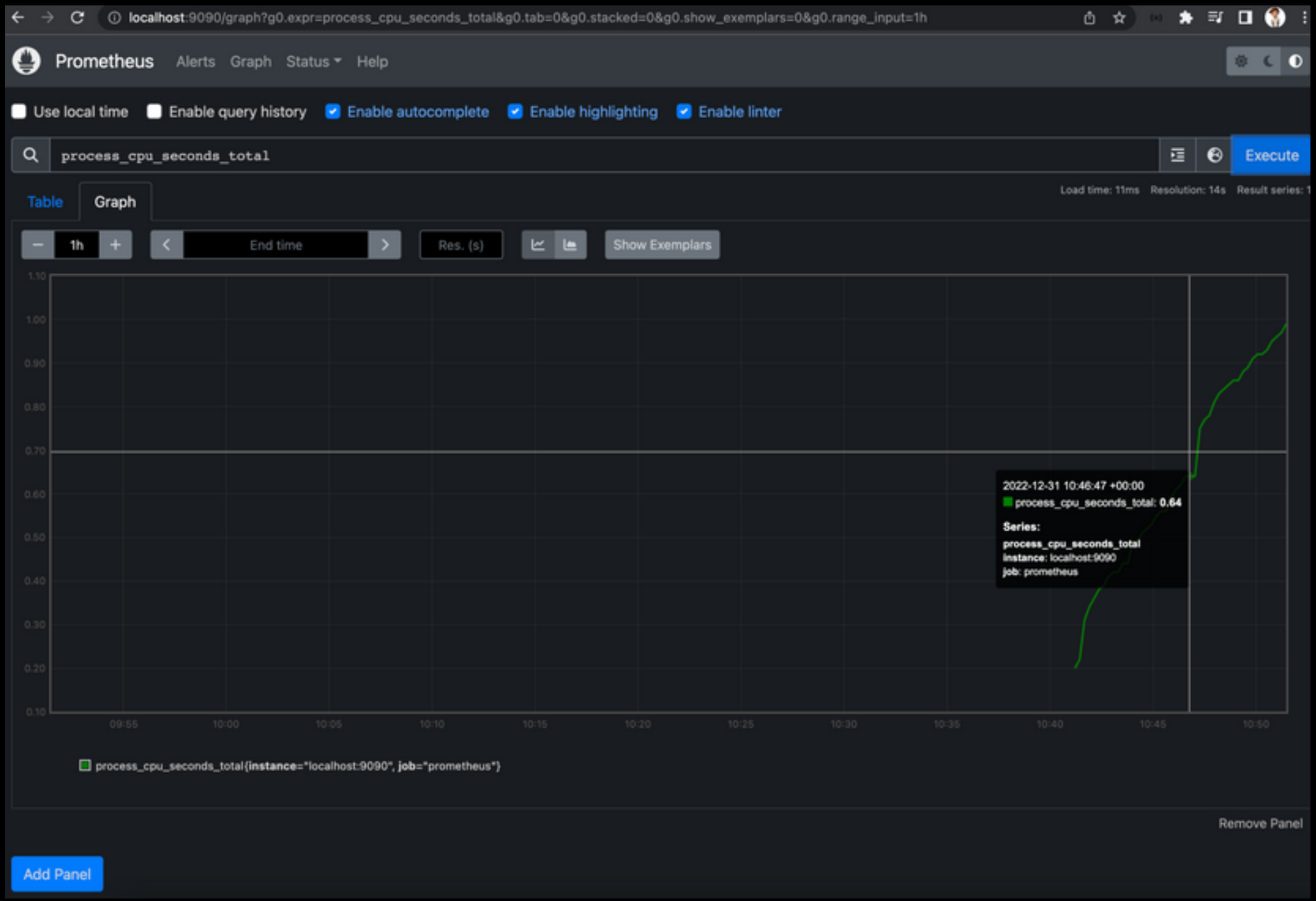
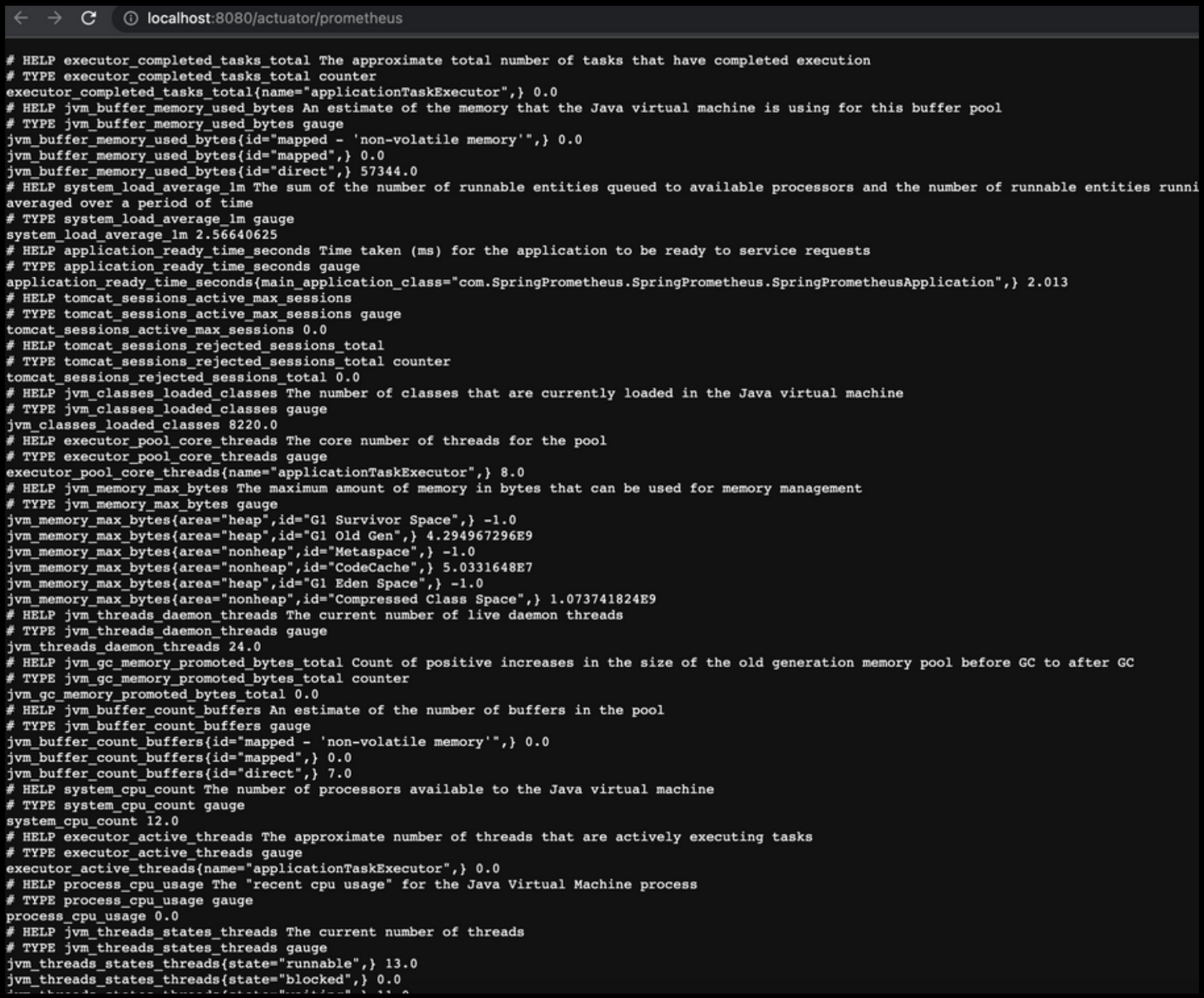
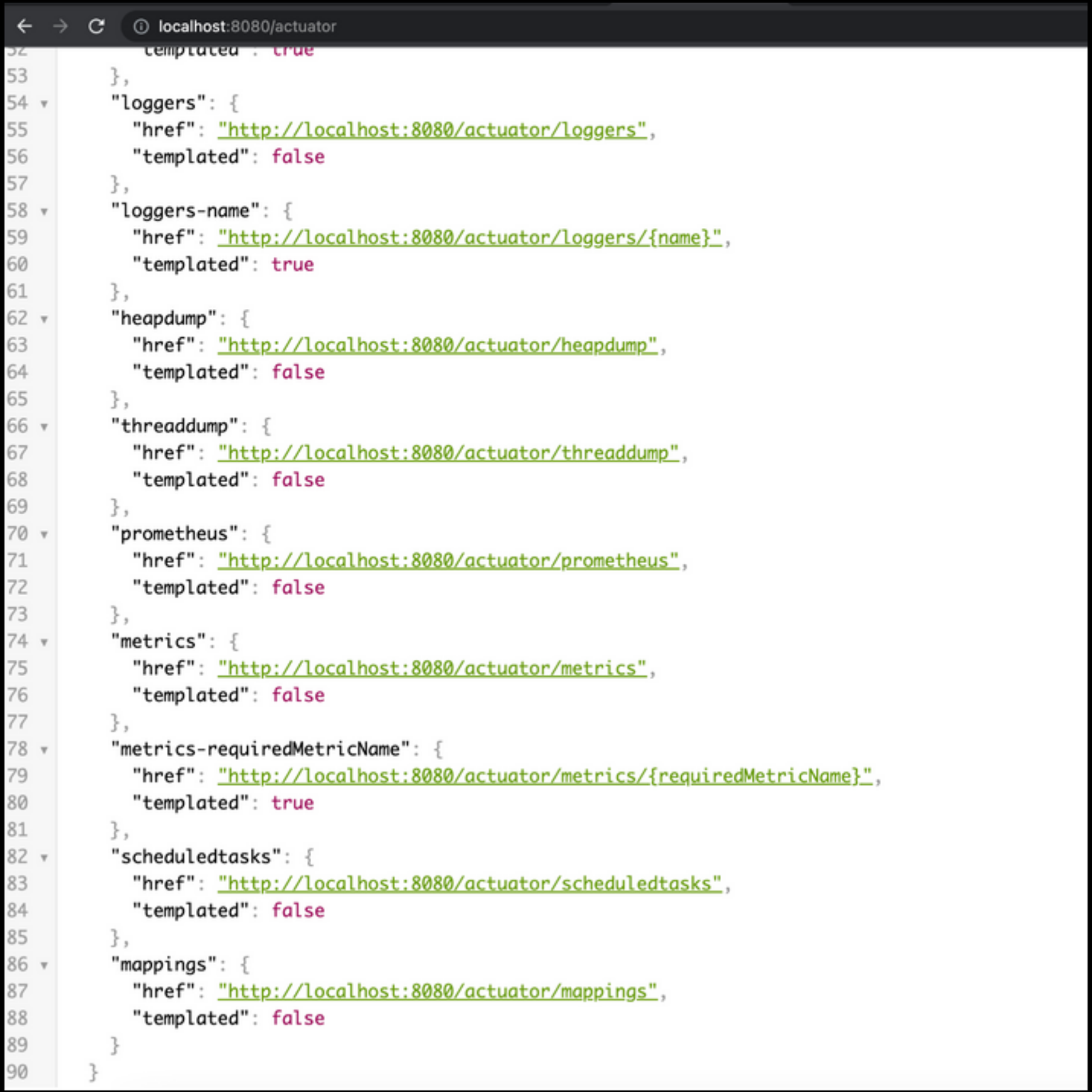
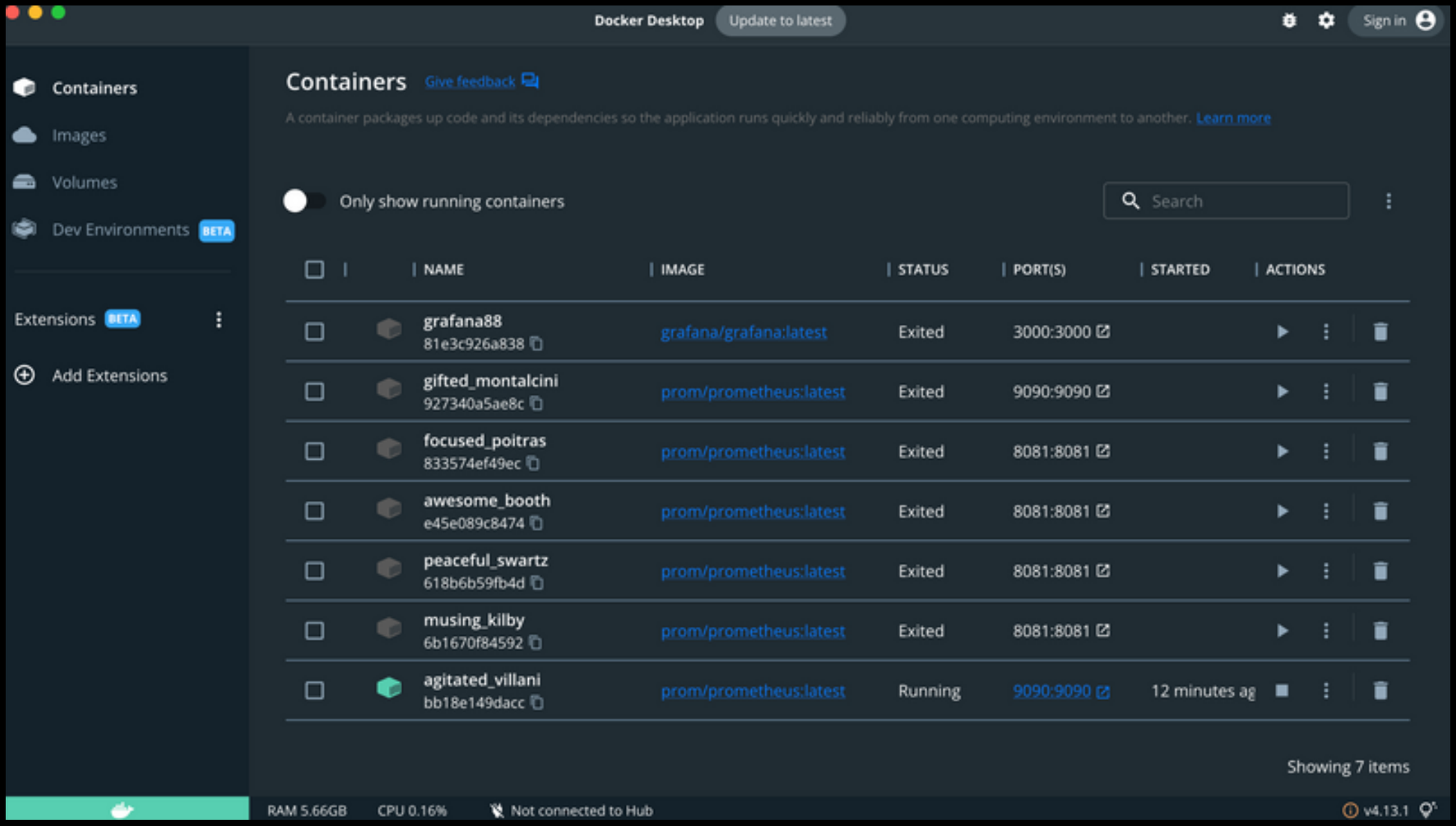



**Spring Application Monitoring Via Prometheus , Docker And Micrometer**


localhost:9090/graph?g0.expr=process_cpu_seconds_total&g0.tab=0&g0.stacked=0&g0.show_exemplars=0&g0.range_input=1h
Prometheus
Alerts
Graph
Status
Help
Use local time
Enable query history
Enable autocomplete
Enable highlighting
Enable linter
process_cpu_seconds_total
Execute
Table
Graph
Load time: 11ms
Resolution: 14s
Result series:
1h
End time
Res. (s)
Show Exemplars
1.10
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
09:55
10:00
10:05
10:10
10:15
10:20
10:25
10:30
10:35
10:40
10:45
10:50
2022-12-31 10:46:47 +00:00
process_cpu_seconds_total: 0.64
Series:
process_cpu_seconds_total
instance: localhost:9090
job: prometheus
process_cpu_seconds_total{instance="localhost:9090", job="prometheus"}
Remove Panel
Add Panel


localhost:8080/actuator/prometheus
# HELP executor_completed_tasks_total The approximate total number of tasks that have completed execution
# TYPE executor_completed_tasks_total counter
executor_completed_tasks_total{name="applicationTaskExecutor",} 0.0
# HELP jvm_buffer_memory_used_bytes An estimate of the memory that the Java virtual machine is using for this buffer pool
# TYPE jvm_buffer_memory_used_bytes gauge
jvm_buffer_memory_used_bytes{id="mapped - 'non-volatile memory'",} 0.0
jvm_buffer_memory_used_bytes{id="mapped",} 0.0
jvm_buffer_memory_used_bytes{id="direct",} 57344.0
# HELP system_load_average_1m The sum of the number of runnable entities queued to available processors and the number of runnable entities runni
averaged over a period of time
# TYPE system_load_average_1m gauge
system_load_average_1m 2.56640625
# HELP application_ready_time_seconds Time taken (ms) for the application to be ready to service requests
# TYPE application_ready_time_seconds gauge
application_ready_time_seconds{main_application_class="com.SpringPrometheus.SpringPrometheus.SpringPrometheusApplication",} 2.013
# HELP tomcat_sessions_active_max_sessions
# TYPE tomcat_sessions_active_max_sessions gauge
tomcat_sessions_active_max_sessions 0.0
# HELP tomcat_sessions_rejected_sessions_total
# TYPE tomcat_sessions_rejected_sessions_total counter
tomcat_sessions_rejected_sessions_total 0.0
# HELP jvm_classes_loaded_classes The number of classes that are currently loaded in the Java virtual machine
# TYPE jvm_classes_loaded_classes gauge
jvm_classes_loaded_classes 8220.0
# HELP executor_pool_core_threads The core number of threads for the pool
# TYPE executor_pool_core_threads gauge
executor_pool_core_threads{name="applicationTaskExecutor",} 8.0
# HELP jvm_memory_max_bytes The maximum amount of memory in bytes that can be used for memory management
# TYPE jvm_memory_max_bytes gauge
jvm_memory_max_bytes{area="heap",id="G1 Survivor Space",} -1.0
jvm_memory_max_bytes{area="heap",id="G1 Old Gen",} 4.294967296E9
jvm_memory_max_bytes{area="nonheap",id="Metaspace",} -1.0
jvm_memory_max_bytes{area="nonheap",id="CodeCache",} 5.0331648E7
jvm_memory_max_bytes{area="heap",id="G1 Eden Space",} -1.0
jvm_memory_max_bytes{area="nonheap",id="Compressed Class Space",} 1.073741824E9
# HELP jvm_threads_daemon_threads The current number of live daemon threads
# TYPE jvm_threads_daemon_threads gauge
jvm_threads_daemon_threads 24.0
# HELP jvm_gc_memory_promoted_bytes_total Count of positive increases in the size of the old generation memory pool before GC to after GC
# TYPE jvm_gc_memory_promoted_bytes_total counter
jvm_gc_memory_promoted_bytes_total 0.0
# HELP jvm_buffer_count_buffers An estimate of the number of buffers in the pool
# TYPE jvm_buffer_count_buffers gauge
jvm_buffer_count_buffers{id="mapped - 'non-volatile memory'",} 0.0
jvm_buffer_count_buffers{id="mapped",} 0.0
jvm_buffer_count_buffers{id="direct",} 7.0
# HELP system_cpu_count The number of processors available to the Java virtual machine
# TYPE system_cpu_count gauge
system_cpu_count 12.0
# HELP executor_active_threads The approximate number of threads that are actively executing tasks
# TYPE executor_active_threads gauge
executor_active_threads{name="applicationTaskExecutor",} 0.0
# HELP process_cpu_usage The "recent cpu usage" for the Java Virtual Machine process
# TYPE process_cpu_usage gauge
process_cpu_usage 0.0
# HELP jvm_threads_states_threads The current number of threads
# TYPE jvm_threads_states_threads gauge
jvm_threads_states_threads{state="runnable",} 13.0
jvm_threads_states_threads{state="blocked",} 0.0


localhost:8080/actuator
templated": true
},
"loggers": {
"href": "http://localhost:8080/actuator/loggers",
"templated": false
},
"loggers-name": {
"href": "http://localhost:8080/actuator/loggers/{name}",
"templated": true
},
"heapdump": {
"href": "http://localhost:8080/actuator/heapdump",
"templated": false
},
"threaddump": {
"href": "http://localhost:8080/actuator/threaddump",
"templated": false
},
"prometheus": {
"href": "http://localhost:8080/actuator/prometheus",
"templated": false
},
"metrics": {
"href": "http://localhost:8080/actuator/metrics",
"templated": false
},
"metrics-requiredMetricName": {
"href": "http://localhost:8080/actuator/metrics/{requiredMetricName}",
"templated": true
},
"scheduledtasks": {
"href": "http://localhost:8080/actuator/scheduledtasks",
"templated": false
},
"mappings": {
"href": "http://localhost:8080/actuator/mappings",
"templated": false
}
}


Docker Desktop
Update to latest
Sign in
Containers
Images
Volumes
Dev Environments
BETA
Extensions
BETA
Add Extensions
Containers
Give feedback
A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more
Only show running containers
Search
NAME
IMAGE
STATUS
PORT(S)
STARTED
ACTIONS
grafana88
81e3c926a838
grafana/grafana:latest
Exited
3000:3000
gifted_montalcini
927340a5ae8c
prom/prometheus:latest
Exited
9090:9090
focused_poitras
833574ef49ec
prom/prometheus:latest
Exited
8081:8081
awesome_booth
e45e089c8474
prom/prometheus:latest
Exited
8081:8081
peaceful_swartz
618b6b59fb4d
prom/prometheus:latest
Exited
8081:8081
musing_kilby
6b1670f84592
prom/prometheus:latest
Exited
8081:8081
agitated_villani
bb18e149dacc
prom/prometheus:latest
Running
9090:9090
12 minutes ag
Showing 7 items
RAM 5.66GB
CPU 0.16%
Not connected to Hub
v4.13.1

- Puppet
- SaltStack

## Using Docker

All Prometheus services are available as Docker images on Quay.io or Docker Hub.

Running Prometheus on Docker is as simple as `docker run -p 9090:9090 prom/prometheus`. This starts Prometheus with a sample configuration and exposes it on port 9090.

The Prometheus image uses a volume to store the actual metrics. For production deployments it is highly recommended to use a named volume to ease managing the data on Prometheus upgrades.

To provide your own configuration, there are several options. Here are two examples.

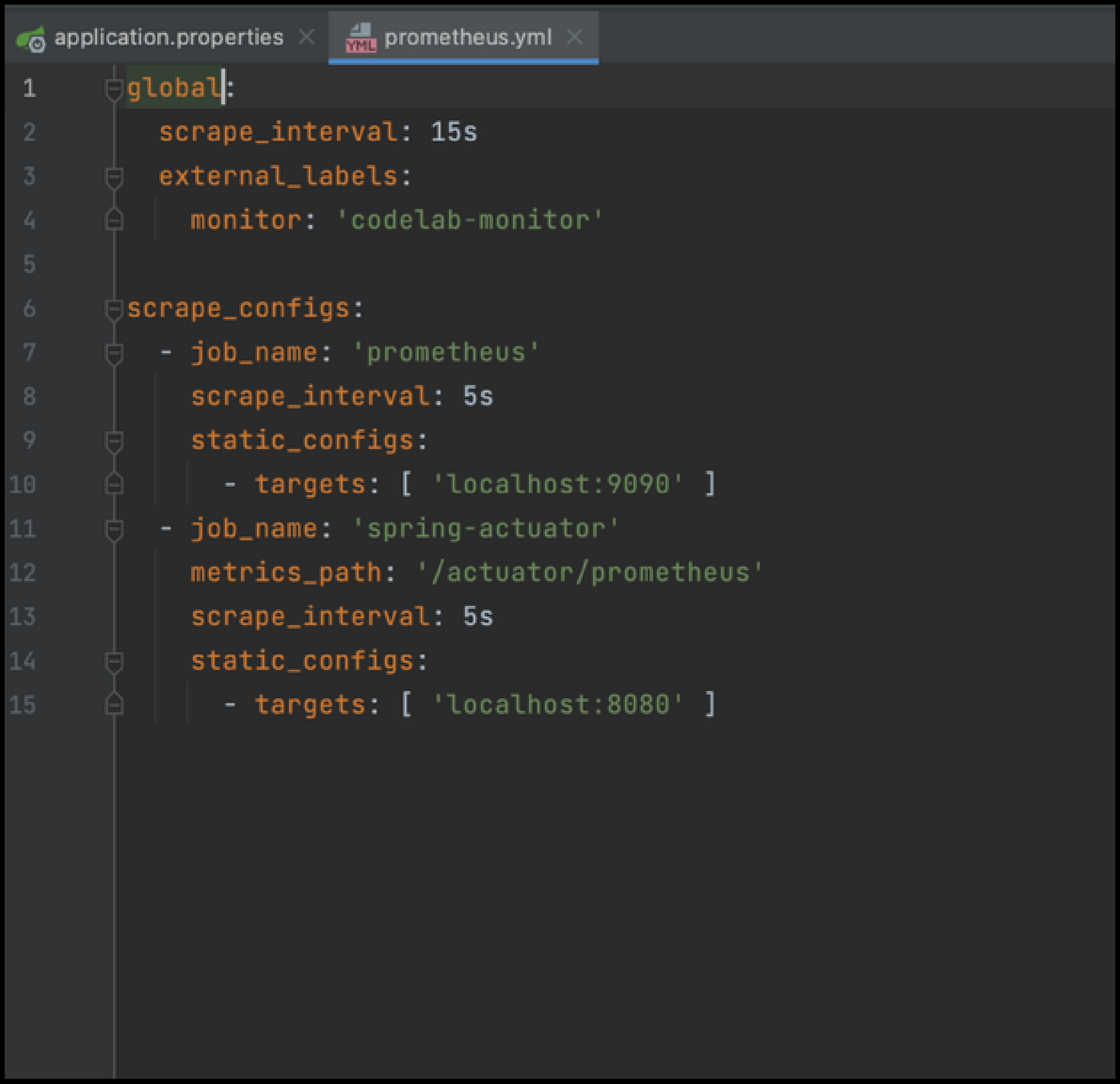
### Volumes & bind-mount

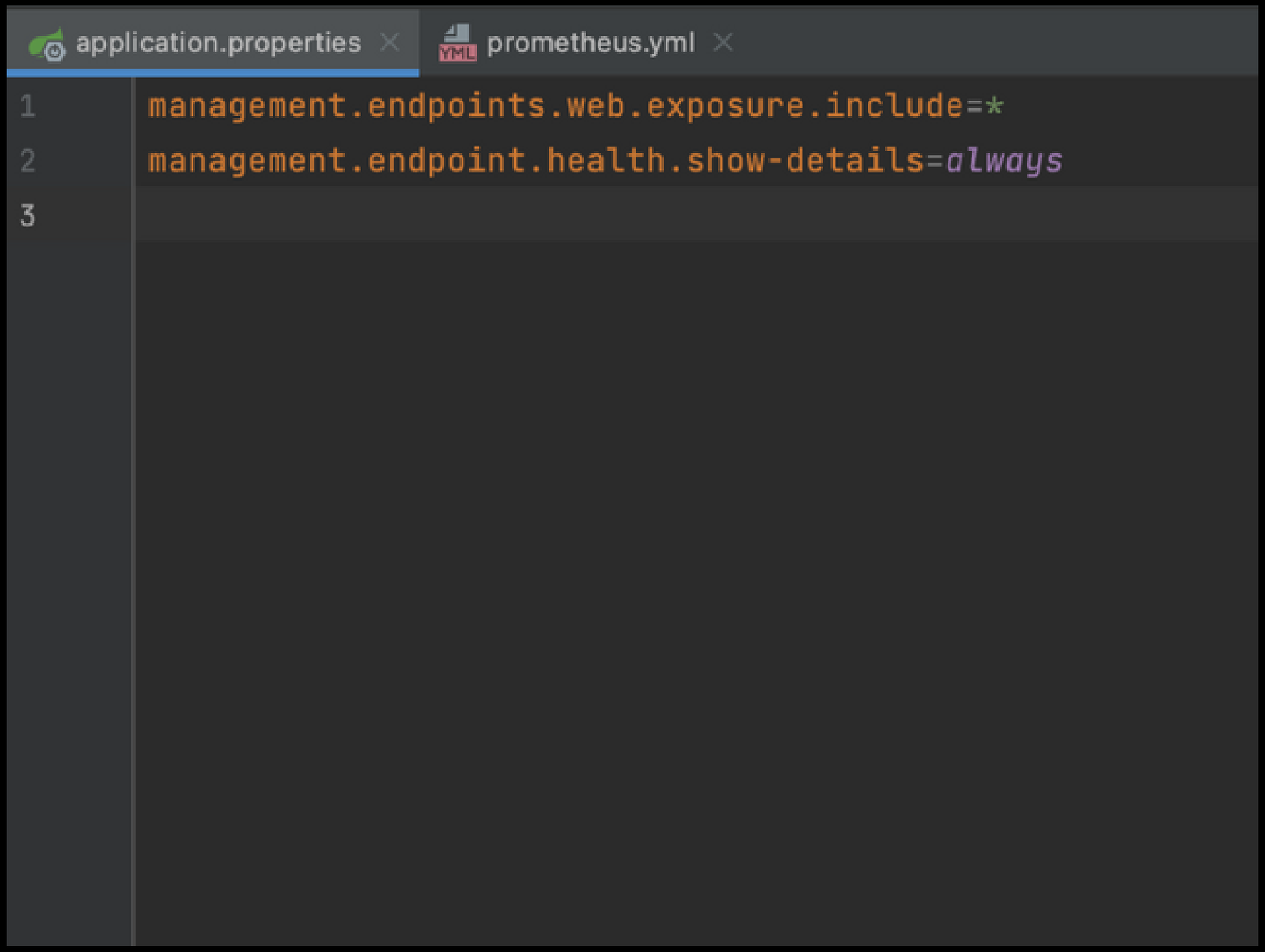
Bind-mount your `prometheus.yml` from the host by running:

```
docker run \
    -p 9090:9090 \
    -v /path/to/prometheus.yml:/etc/prometheus/prometheus.yml \
    prom/prometheus
```

Or bind-mount the directory containing `prometheus.yml` onto `/etc/prometheus` by running:

```
docker run \
    -p 9090:9090 \
    -v /path/to/config:/etc/prometheus \
    prom/prometheus
```


application.properties
prometheus.yml
global:
scrape_interval: 15s
external_labels:
monitor: 'codelab-monitor'
scrape_configs:
- job_name: 'prometheus'
scrape_interval: 5s
static_configs:
- targets: [ 'localhost:9090' ]
- job_name: 'spring-actuator'
metrics_path: '/actuator/prometheus'
scrape_interval: 5s
static_configs:
- targets: [ 'localhost:8080' ]


application.properties
prometheus.yml
1 management.endpoints.web.exposure.include=*
2 management.endpoint.health.show-details=always
3

```xml
<dependencies>
        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-actuator</artifactId>
        </dependency>
        <dependency>
                <groupId>io.micrometer</groupId>
                <artifactId>micrometer-registry-prometheus</artifactId>
        </dependency>
        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-test</artifactId>
                <scope>test</scope>
        </dependency>
</dependencies>
```