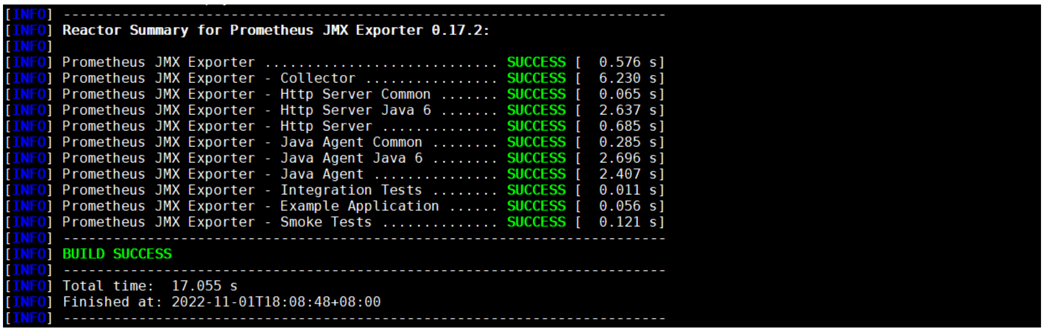
jmx-exporter配置说明

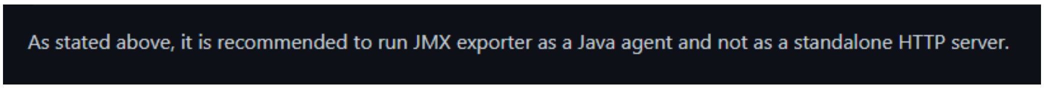
一、编译安装jmx-exporter

```
[root@zhoufr-kafka-node01 sd_config]# cd /root/
[root@zhoufr-kafka-node01 jmx_exporter-parent-0.17.2]# ls -lh
total 96K
-rw-rw-r-- 1 root root  152 Sep 22 05:35 CODE_OF_CONDUCT.md
drwxrwxr-x 4 root root   71 Oct 31 18:33 collector
-rw-rw-r-- 1 root root  560 Sep 22 05:35 CONTRIBUTING.md
drwxrwxr-x 2 root root 4.0K Nov  1 11:00 example_configs
drwxrwxr-x 4 root root   96 Sep 22 05:35 integration_tests
drwxrwxr-x 4 root root  105 Nov  1 18:08 jmx_prometheus_httpserver
drwxrwxr-x 4 root root   71 Oct 31 18:33 jmx_prometheus_httpserver_common
drwxrwxr-x 4 root root  105 Nov  1 18:08 jmx_prometheus_httpserver_java6
drwxrwxr-x 3 root root  112 Nov  1 18:08 jmx_prometheus_javaagent
drwxrwxr-x 4 root root   71 Oct 31 18:37 jmx_prometheus_javaagent_common
drwxrwxr-x 3 root root   94 Nov  1 18:08 jmx_prometheus_javaagent_java6
-rw-rw-r-- 1 root root  12K Sep 22 05:35 LICENSE
-rw-rw-r-- 1 root root  836 Sep 22 05:35 MAINTAINER_NOTES.md
-rw-rw-r-- 1 root root   84 Sep 22 05:35 MAINTAINERS.md
-rwxrwxr-x 1 root root 9.9K Sep 22 05:35 mvnw
-rw-rw-r-- 1 root root 6.5K Sep 22 05:35 mvnw.cmd
-rw-rw-r-- 1 root root  165 Sep 22 05:35 NOTICE
-rw-rw-r-- 1 root root  13K Sep 22 05:35 pom.xml
-rw-rw-r-- 1 root root 9.9K Sep 22 05:35 README.md
-rwxrwxr-x 1 root root  510 Sep 22 05:35 run_sample_httpserver.sh
-rw-rw-r-- 1 root root  172 Sep 22 05:35 SECURITY.md
-rw-rw-r-- 1 root root  353 Sep 22 05:35 version-rules.xml
[root@zhoufr-kafka-node01 jmx_exporter-parent-0.17.2]# ./mvnw package
......
```



```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Prometheus JMX Exporter 0.17.2:
[INFO]
[INFO] Prometheus JMX Exporter ............................ SUCCESS [  0.576 s]
[INFO] Prometheus JMX Exporter - Collector ................ SUCCESS [  6.230 s]
[INFO] Prometheus JMX Exporter - Http Server Common ....... SUCCESS [  0.065 s]
[INFO] Prometheus JMX Exporter - Http Server Java 6 ....... SUCCESS [  2.637 s]
[INFO] Prometheus JMX Exporter - Http Server .............. SUCCESS [  0.685 s]
[INFO] Prometheus JMX Exporter - Java Agent Common ........ SUCCESS [  0.285 s]
[INFO] Prometheus JMX Exporter - Java Agent Java 6 ........ SUCCESS [  2.696 s]
[INFO] Prometheus JMX Exporter - Java Agent ............... SUCCESS [  2.407 s]
[INFO] Prometheus JMX Exporter - Integration Tests ........ SUCCESS [  0.011 s]
[INFO] Prometheus JMX Exporter - Example Application ...... SUCCESS [  0.056 s]
[INFO] Prometheus JMX Exporter - Smoke Tests .............. SUCCESS [  0.121 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  17.055 s
[INFO] Finished at: 2022-11-01T18:08:48+08:00
[INFO] ------------------------------------------------------------------------
```

编译完成后，你会在`jmx_prometheus_javaagent/target/`中发现编译好的jar包（官方推荐使用agent的试， 而非http的 export）所以我们这里以agent为例



As stated above, it is recommended to run JMX exporter as a Java agent and not as a standalone HTTP server.

```
[root@zhoufr-kafka-node01 jmx_exporter-parent-0.17.2]# ls -lh jmx_prometheus_javaagent/target/
total 2.3M
drwxr-xr-x 4 root root  294 Nov  1 18:08 apidocs
drwxr-xr-x 2 root root   71 Nov  1 18:08 javadoc-bundle-options
-rw-r--r-- 1 root root 527K Nov  1 18:08 jmx_prometheus_javaagent-0.17.2.jar
-rw-r--r-- 1 root root  26K Nov  1 18:08 jmx_prometheus_javaagent-0.17.2-javadoc.jar
-rw-r--r-- 1 root root 791K Nov  1 18:08 jmx_prometheus_javaagent-0.17.2-sources.jar
drwxr-xr-x 2 root root   28 Oct 31 18:37 maven-archiver
-rw-r--r-- 1 root root 3.9K Oct 31 18:37 maven-javadoc-plugin-stale-data.txt
-rw-r--r-- 1 root root 527K Oct 31 18:37 original-jmx_prometheus_javaagent-0.17.2.jar
-rw-r--r-- 1 root root 419K Oct 31 18:37 original-jmx_prometheus_javaagent-0.17.2-sources.jar
```

将jar包拷入目标目录（/opt/Apps/jmx_prometheus_javaagent/）， 配置文件可以简单的配置成：

```
rules:
- pattern: ".*"
```

也可以从源码目录example_configs中复制， 对应的配置文件， 比如：

```
[root@zhoufr-kafka-node01 jmx_prometheus_agent]# ls  -lh /root/jmx_exporter-parent-0.17.2/example_configs/
total 72K
-rw-rw-r-- 1 root root 1.2K Sep 22 05:35 activemq.yml
-rw-rw-r-- 1 root root  789 Sep 22 05:35 artemis-2.yml
-rw-rw-r-- 1 root root 1.9K Sep 22 05:35 cassandra.yml
-rw-rw-r-- 1 root root 2.4K Sep 22 05:35 flink.yml
-rw-rw-r-- 1 root root   76 Sep 22 05:35 httpserver_sample_config.yml
-rw-rw-r-- 1 root root 2.8K Sep 22 05:35 kafka-0-8-2.yml
-rw-rw-r-- 1 root root 3.3K Sep 22 05:35 kafka-2_0_0.yml
-rw-rw-r-- 1 root root 4.4K Sep 22 05:35 kafka-connect.yml
-rw-rw-r-- 1 root root 2.0K Sep 22 05:35 kafka-pre0-8-2.yml
-rw-rw-r-- 1 root root 4.5K Sep 22 05:35 spark-3-0.yml
-rw-rw-r-- 1 root root 4.1K Sep 22 05:35 spark.yml
-rw-rw-r-- 1 root root 1.2K Sep 22 05:35 tomcat.yml
-rw-rw-r-- 1 root root 2.0K Sep 22 05:35 weblogic.yml
-rw-rw-r-- 1 root root 1.6K Sep 22 05:35 wildfly-10.yaml
-rw-rw-r-- 1 root root 1.3K Sep 22 05:35 zookeeper.yaml
```

在相应的应用启动参数中增加javaagent配置：

```
java -javaagent:./jmx_prometheus_javaagent-0.17.2.jar=12345:config.yaml -jar yourJar.jar
```

我这边的应用是一个kafka2集群， 下面是增加agnet的配置说明

1. zookeeper

zookeeper需要在bin目录中的zkEnv.sh中SERVER_JVMFLAGS参数追加， 如下所示：

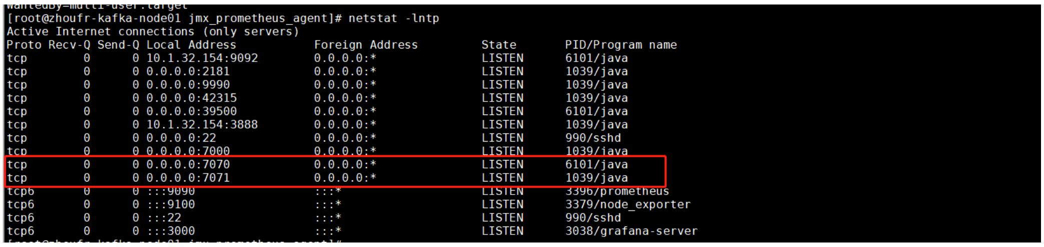
```
export SERVER_JVMFLAGS="-Xmx${ZK_SERVER_HEAP}m $SERVER_JVMFLAGS -javaagent:/opt/Apps/jmx_prometheus_agent/jmx_prometheus_javaagent-0.17.2.jar=7071:/opt/Apps/jmx_prometheus_agent/zookeeper.yaml"
```

2. kafka

kafka直接修改systemd的守护进程文件kafka.service

```
[Unit]
Description=Apache Kafka server (broker)
Documentation=http://kafka.apache.org/documentation.html
Requires=network.target
After=network.target zk.service

[Service]
Type=forking
User=dominos-op
Group=dominos-op
Environment=JAVA_HOME=/usr/local/java/jdk1.8.0_112
Environment="KAFKA_OPTS=-javaagent:/opt/Apps/jmx_prometheus_agent/jmx_prometheus_javaagent-0.17.2.jar=7070:/opt/Apps/jmx_prometheus_agent/config.yaml"
ExecStart=/opt/Apps/kafka/bin/kafka-server-start.sh  -daemon /opt/Apps/kafka/config/server.properties
Restart=on-failure
LimitNOFILE=16384:163840

[Install]
WantedBy=multi-user.target
```

修改完成后，重新启动两个服务， 然后使用ps –aux | grep javaagent 查看是否生效或者使用netstat也可以检测：



```
WantedBy=multi-user.target
[root@zhoufr-kafka-node01 jmx_prometheus_agent]# netstat -lntp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 10.1.32.154:9092        0.0.0.0:*               LISTEN      6101/java
tcp        0      0 0.0.0.0:2181            0.0.0.0:*               LISTEN      1039/java
tcp        0      0 0.0.0.0:9990            0.0.0.0:*               LISTEN      1039/java
tcp        0      0 0.0.0.0:42315           0.0.0.0:*               LISTEN      1039/java
tcp        0      0 0.0.0.0:39500           0.0.0.0:*               LISTEN      6101/java
tcp        0      0 10.1.32.154:3888        0.0.0.0:*               LISTEN      1039/java
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      990/sshd
tcp        0      0 0.0.0.0:7000            0.0.0.0:*               LISTEN      1039/java
tcp        0      0 0.0.0.0:7070            0.0.0.0:*               LISTEN      6101/java
tcp        0      0 0.0.0.0:7071            0.0.0.0:*               LISTEN      1039/java
tcp6       0      0 :::9090                 :::*                    LISTEN      3396/prometheus
tcp6       0      0 :::9100                 :::*                    LISTEN      3379/node_exporter
tcp6       0      0 :::22                   :::*                    LISTEN      990/sshd
tcp6       0      0 :::3000                 :::*                    LISTEN      3038/grafana-server
```

也可以通过http://ip:7070/metrics 进行检查， 至此jmx的编译完成。

二、修改prometheus配置

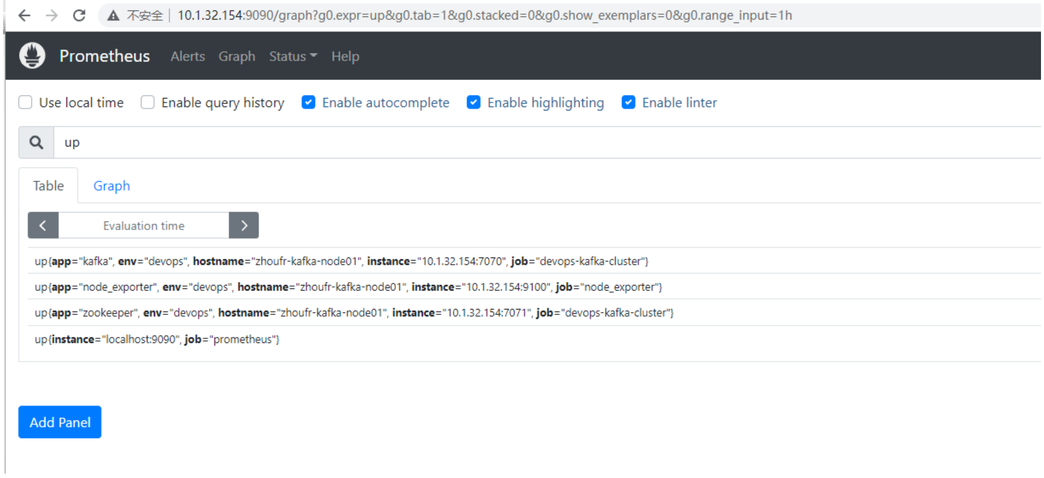
全局配置

```
[root@zhoufr-kafka-node01 prometheus]# cat prometheus.yml
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Defaul
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'ev
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scra
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "devops-kafka-cluster"
    file_sd_configs:
    - files: ["/opt/Apps/prometheus/sd_config/jvm.yml"]

  - job_name: "node_exporter"
    file_sd_configs:
    - files: ["/opt/Apps/prometheus/sd_config/linux.yml"][root@zhoufr-kafka-
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Defaul
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'ev
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scra
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "devops-kafka-cluster"
    file_sd_configs:
    - files: ["/opt/Apps/prometheus/sd_config/jvm.yml"]

  - job_name: "node_exporter"
    file_sd_configs:
    - files: ["/opt/Apps/prometheus/sd_config/linux.yml"]
```

sd_config下面的jvm.yml配置：

```
[root@zhoufr-kafka-node01 prometheus]# cat /opt/Apps/prometheus/sd_config/jv
- targets:
  - 10.1.32.154:7071
  labels:
    env: devops
    app: zookeeper
    hostname: zhoufr-kafka-node01


- targets:
  - 10.1.32.154:7070
  labels:
    env: devops
    app: kafka
    hostname: zhoufr-kafka-node01
```

设置完成后，在http://ip:9090/ 可以查询到即可：


不安全 | 10.1.32.154:9090/graph?g0.expr=up&g0.tab=1&g0.stacked=0&g0.show_exemplars=0&g0.range_input=1h
Prometheus
Alerts
Graph
Status
Help
Table
Graph
Evaluation time
Add Panel

☐ Use local time ☐ Enable query history ☑ Enable autocomplete ☑ Enable highlighting ☑ Enable linter

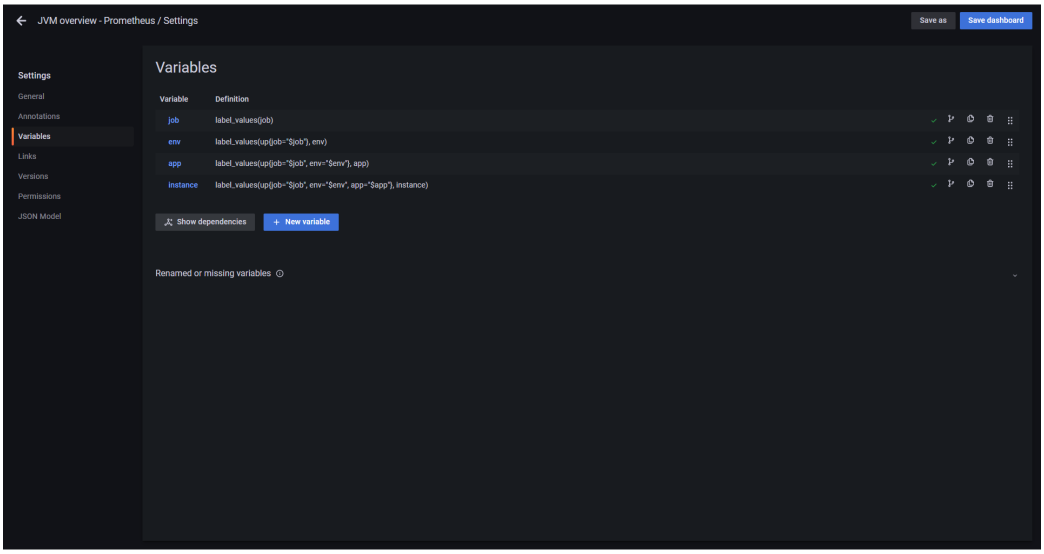
```
up
```

```
up{app="kafka", env="devops", hostname="zhoufr-kafka-node01", instance="10.1.32.154:7070", job="devops-kafka-cluster"}
up{app="node_exporter", env="devops", hostname="zhoufr-kafka-node01", instance="10.1.32.154:9100", job="node_exporter"}
up{app="zookeeper", env="devops", hostname="zhoufr-kafka-node01", instance="10.1.32.154:7071", job="devops-kafka-cluster"}
up{instance="localhost:9090", job="prometheus"}
```

至此Prometheus可以成功抓取相应的jvm指标了

三、grafana自定义配置

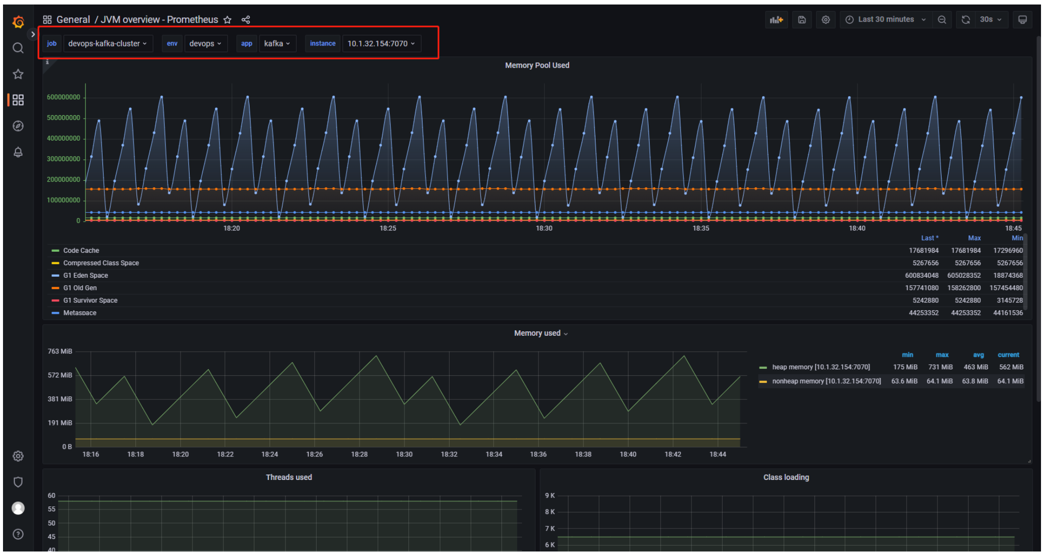
这里需要导入一个基础模板ID：3066, 这个模板是一个基础性的模板， 更多的性能监控，需要我们根据需要来定义， 因为我们在prometheus里配置文件里面定义了相应的labels所以，我们导入模板后， 需要定义一些模板变量，来自定义我们的dashboard， 比如：



Variables

Variable	Definition
job	label_values(job)
env	label_values(up{job="$job"}, env)
app	label_values(up{job="$job", env="$env"}, app)
instance	label_values(up{job="$job", env="$env", app="$app"}, instance)

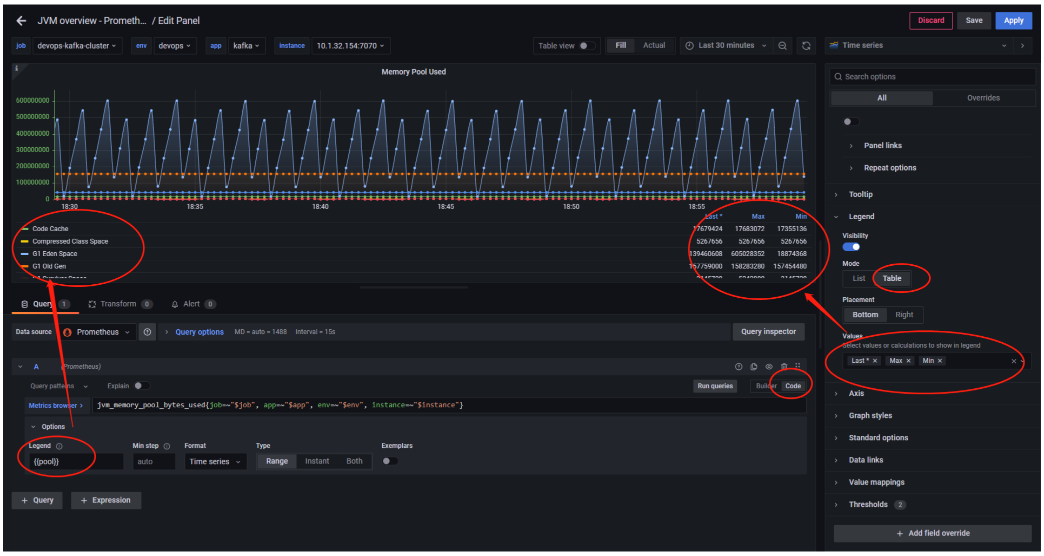
定义完成后，我们可以通过模板变量做一些细致的过滤



	Last *	Max	Min
Code Cache	17681984	17681984	17296960
Compressed Class Space	5267656	5267656	5267656
G1 Eden Space	600834048	605028352	18874368
G1 Old Gen	157741080	158262800	157454480
G1 Survivor Space	5242880	5242880	3145728
Metaspace	44253352	44253352	44161536

	min	max	avg	current
heap memory [10.1.32.154:7070]	175 MiB	731 MiB	463 MiB	562 MiB
nonheap memory [10.1.32.154:7070]	63.6 MiB	64.1 MiB	63.8 MiB	64.1 MiB

grafana的自定义



	Last *	Max	Min
Code Cache	17679424	17683072	17355136
Compressed Class Space	5267656	5267656	5267656
G1 Eden Space	139460608	605028352	18874368
G1 Old Gen	157759000	158283280	157454480

Data source Prometheus
Query options MD = auto = 1488 Interval = 15s

A (Prometheus)

```
jvm_memory_pool_bytes_used{job=~"$job", app=~"$app", env=~"$env", instance=~"$instance"}
```

Options
Legend {{pool}}
Min step auto
Format Time series
Type Range

自定义有四个地方必须要指定， 上图中已经框出，选择code可以使用习惯的pmsql编写查询语句，legend指定展示多少内容， values也是图标中要展示的最新值 ， 最大值和最小值，table的展示会让你感觉到无比的清爽。