DATADOG

Datadog Custom Metrics DogStatsD

January 31, 2022

Agenda

- What is DogStatsd?
    - Introduction to Datadog Custom Metrics
    - Metrics Summary / Explorer
- Custom Metrics Demo
    - Pull Custom Metrics from Database Table using Java
- Dashboarding Tips and Tricks
    - Custom Metrics Dashboard with Template Variables

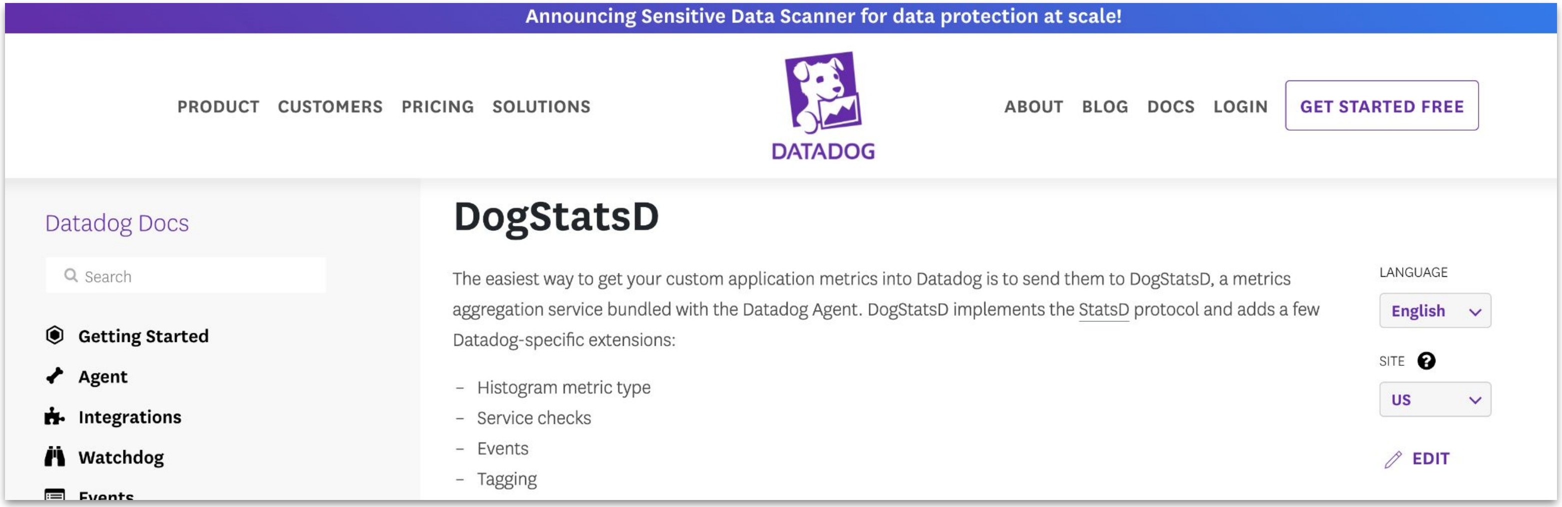
DATADOG

DogStatsD

StatsD was introduced by Etsy in 2011, it has become a mainstay of infrastructure monitoring.

Datadog has extended DogStatsD to support tagging metrics with key-value pairs.

DATADOG



Announcing Sensitive Data Scanner for data protection at scale!

DogStatsD

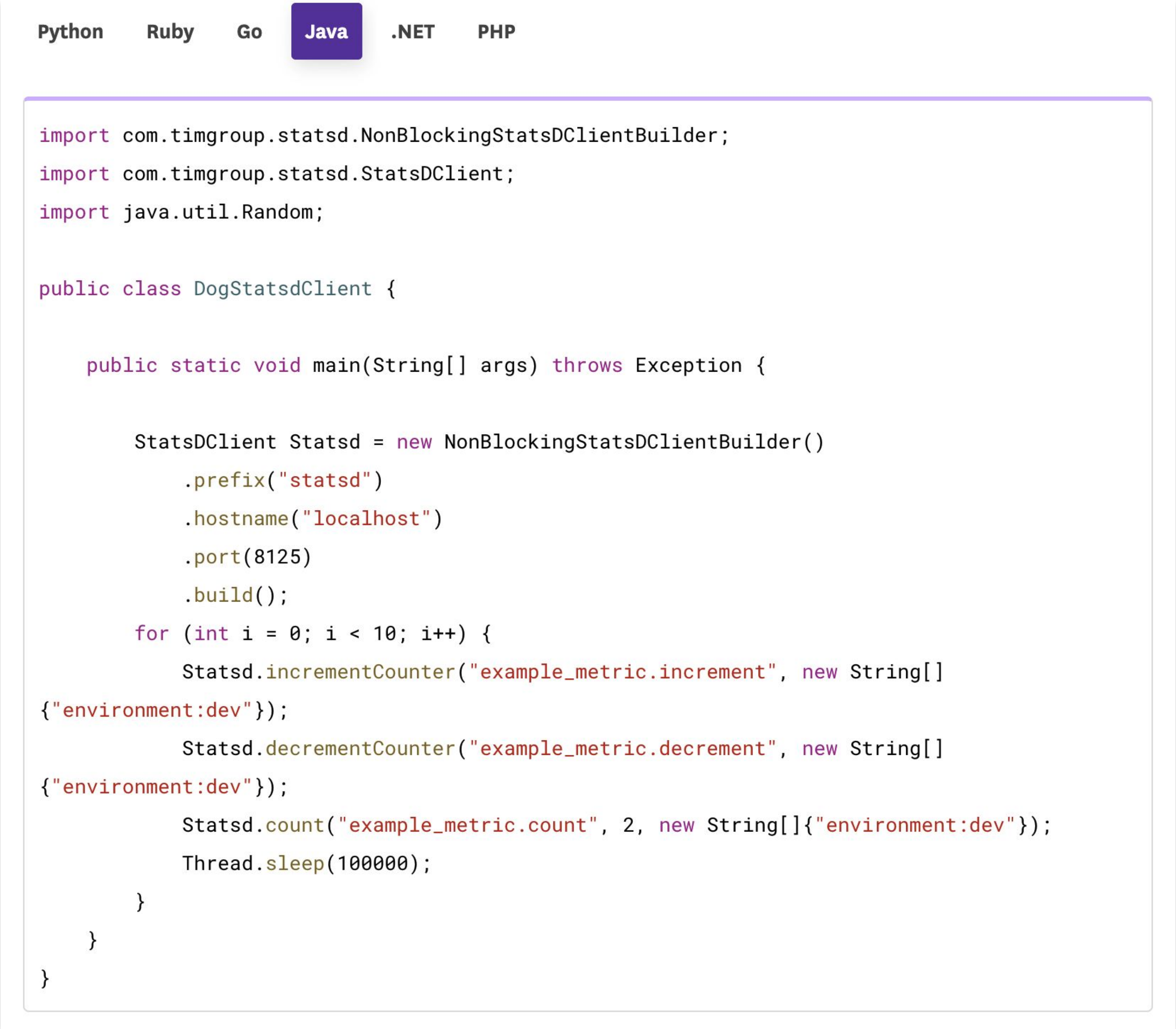
The easiest way to get your custom application metrics into Datadog is to send them to DogStatsD, a metrics aggregation service bundled with the Datadog Agent. DogStatsD implements the StatsD protocol and adds a few Datadog-specific extensions:

- Histogram metric type
- Service checks
- Events
- Tagging

As dev and ops teams rely more and more on containerized microservices, they have pushed the limits of StatsD’s design. One major shortcoming is that StatsD has no built-in support for tagging your metrics with key-value pairs. DogStatsD solves this problem.

DogStatsD

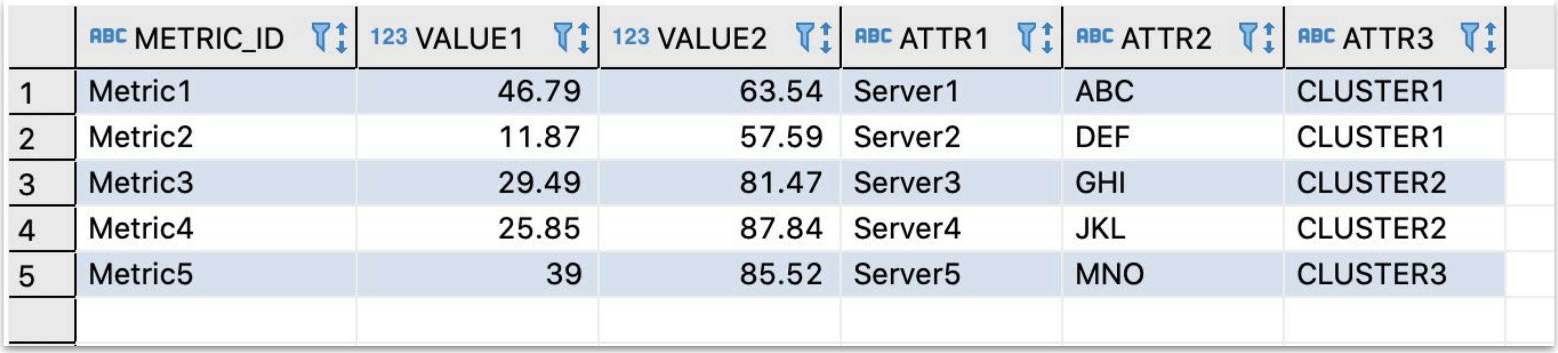
Datadog Sample Application



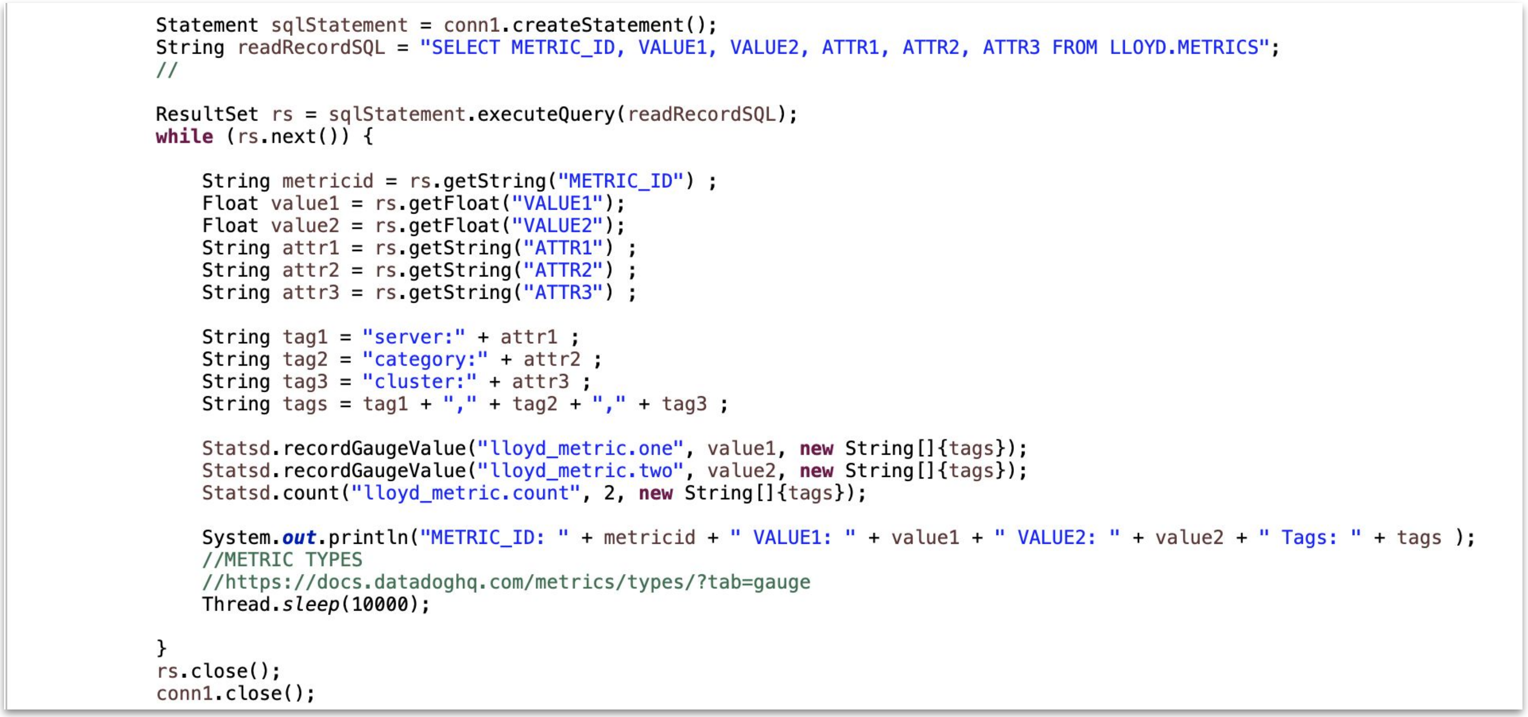
```java
import com.timgroup.statsd.NonBlockingStatsDClientBuilder;
import com.timgroup.statsd.StatsDClient;
import java.util.Random;

public class DogStatsdClient {

    public static void main(String[] args) throws Exception {

        StatsDClient Statsd = new NonBlockingStatsDClientBuilder()
            .prefix("statsd")
            .hostname("localhost")
            .port(8125)
            .build();
        for (int i = 0; i < 10; i++) {
            Statsd.incrementCounter("example_metric.increment", new String[]
{"environment:dev"});
            Statsd.decrementCounter("example_metric.decrement", new String[]
{"environment:dev"});
            Statsd.count("example_metric.count", 2, new String[]{"environment:dev"});
            Thread.sleep(100000);
        }
    }
}
```

DogStatsD

Lloyd's sample of pulling custom metrics from a database table



	ABC METRIC_ID	123 VALUE1	123 VALUE2	ABC ATTR1	ABC ATTR2	ABC ATTR3
1	Metric1	46.79	63.54	Server1	ABC	CLUSTER1
2	Metric2	11.87	57.59	Server2	DEF	CLUSTER1
3	Metric3	29.49	81.47	Server3	GHI	CLUSTER2
4	Metric4	25.85	87.84	Server4	JKL	CLUSTER2
5	Metric5	39	85.52	Server5	MNO	CLUSTER3

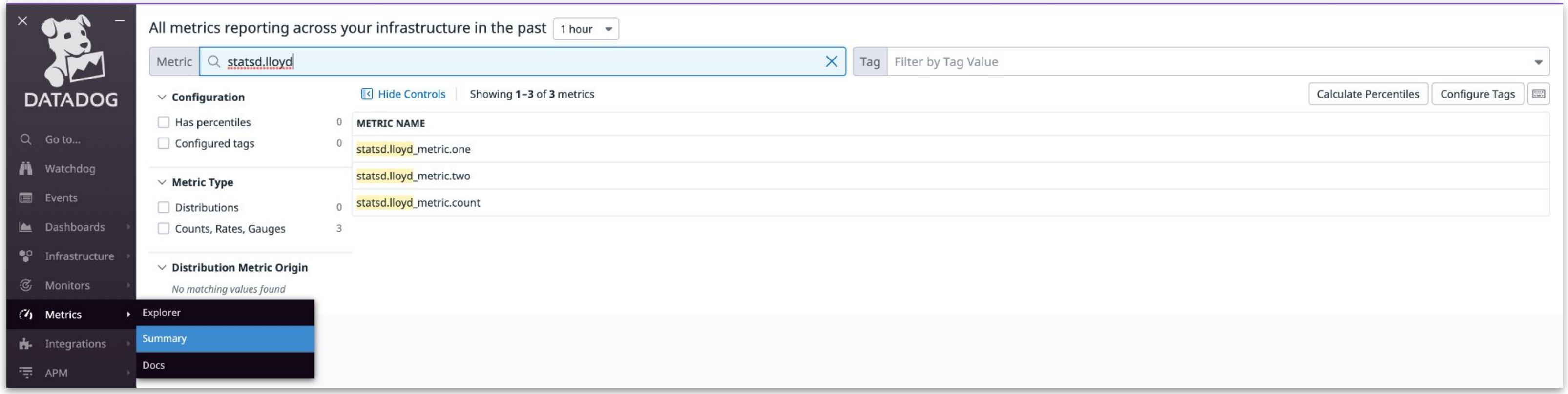


```
Statement sqlStatement = conn1.createStatement();
String readRecordSQL = "SELECT METRIC_ID, VALUE1, VALUE2, ATTR1, ATTR2, ATTR3 FROM LLOYD.METRICS";
//

ResultSet rs = sqlStatement.executeQuery(readRecordSQL);
while (rs.next()) {

    String metricid = rs.getString("METRIC_ID") ;
    Float value1 = rs.getFloat("VALUE1");
    Float value2 = rs.getFloat("VALUE2");
    String attr1 = rs.getString("ATTR1") ;
    String attr2 = rs.getString("ATTR2") ;
    String attr3 = rs.getString("ATTR3") ;

    String tag1 = "server:" + attr1 ;
    String tag2 = "category:" + attr2 ;
    String tag3 = "cluster:" + attr3 ;
    String tags = tag1 + "," + tag2 + "," + tag3 ;

    Statsd.recordGaugeValue("lloyd_metric.one", value1, new String[]{tags});
    Statsd.recordGaugeValue("lloyd_metric.two", value2, new String[]{tags});
    Statsd.count("lloyd_metric.count", 2, new String[]{tags});

    System.out.println("METRIC_ID: " + metricid + " VALUE1: " + value1 + " VALUE2: " + value2 + " Tags: " + tags );
    //METRIC TYPES
    //https://docs.datadoghq.com/metrics/types/?tab=gauge
    Thread.sleep(10000);

}
rs.close();
conn1.close();
```

DATADOG

DogStatsD

Datadog Metrics Summary



All metrics reporting across your infrastructure in the past 1 hour

Metric: statsd.lloyd — Tag: Filter by Tag Value

Showing 1–3 of 3 metrics

Configuration

- □ Has percentiles 0
- □ Configured tags 0

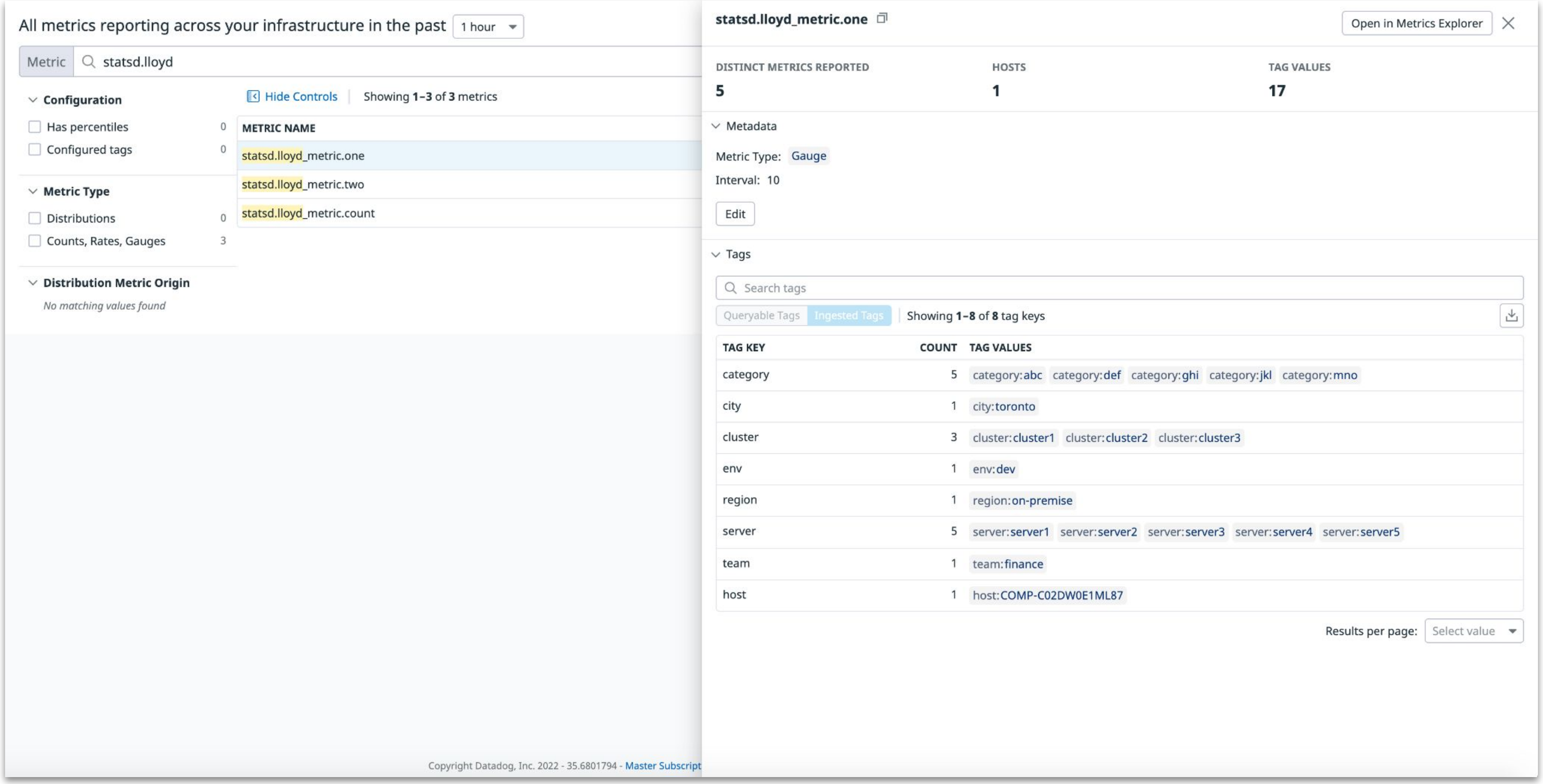
Metric Type

- □ Distributions 0
- □ Counts, Rates, Gauges 3

Distribution Metric Origin

No matching values found

METRIC NAME
statsd.lloyd_metric.one
statsd.lloyd_metric.two
statsd.lloyd_metric.count



All metrics reporting across your infrastructure in the past 1 hour

Metric: statsd.lloyd

Showing 1–3 of 3 metrics

Configuration

- □ Has percentiles 0
- □ Configured tags 0

Metric Type

- □ Distributions 0
- □ Counts, Rates, Gauges 3

Distribution Metric Origin

No matching values found

METRIC NAME
statsd.lloyd_metric.one
statsd.lloyd_metric.two
statsd.lloyd_metric.count

statsd.lloyd_metric.one

DISTINCT METRICS REPORTED	HOSTS	TAG VALUES
5	1	17

Metadata

Metric Type: Gauge

Interval: 10

Tags

Search tags

Showing 1–8 of 8 tag keys

TAG KEY	COUNT	TAG VALUES
category	5	category:abc category:def category:ghi category:jkl category:mno
city	1	city:toronto
cluster	3	cluster:cluster1 cluster:cluster2 cluster:cluster3
env	1	env:dev
region	1	region:on-premise
server	5	server:server1 server:server2 server:server3 server:server4 server:server5
team	1	team:finance
host	1	host:COMP-C02DW0E1ML87

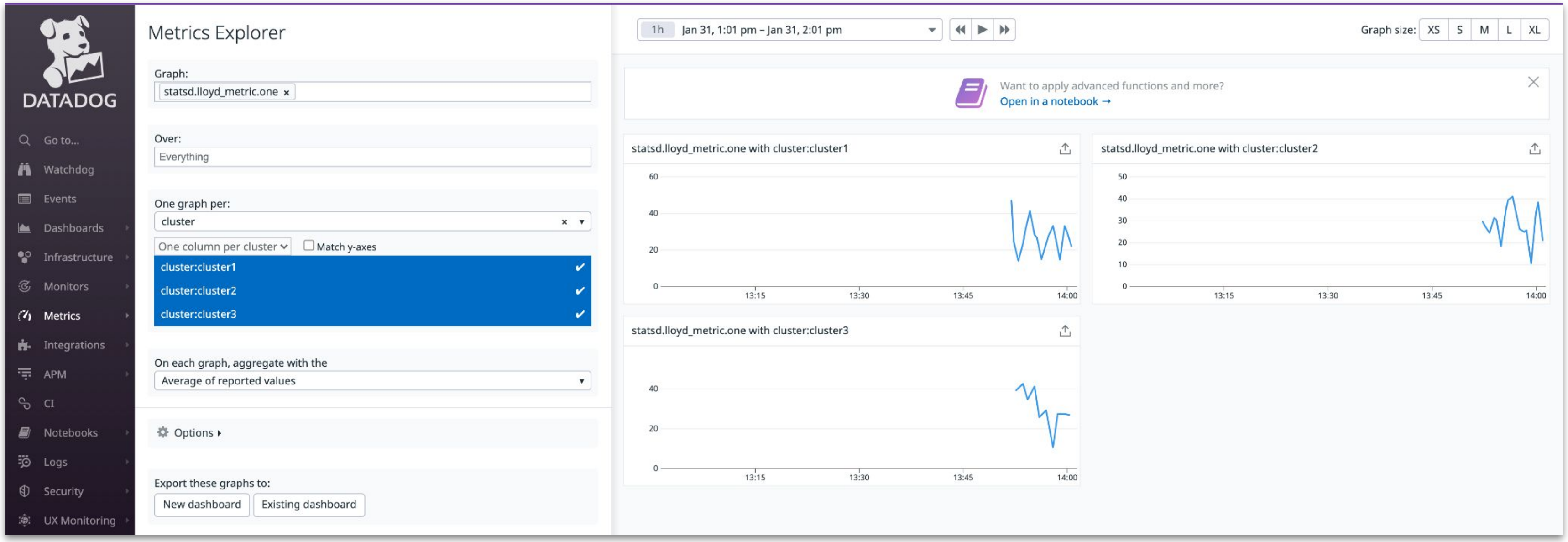
Results per page: Select value

Copyright Datadog, Inc. 2022 - 35.6801794 - Master Subscript

DATADOG

DogStatsD

Datadog Metrics Explorer



Metrics Explorer

Graph: statsd.lloyd_metric.one ×

Over: Everything

One graph per: cluster

One column per cluster

□ Match y-axes

cluster:cluster1 ✔

cluster:cluster2 ✔

cluster:cluster3 ✔

On each graph, aggregate with the Average of reported values

Options

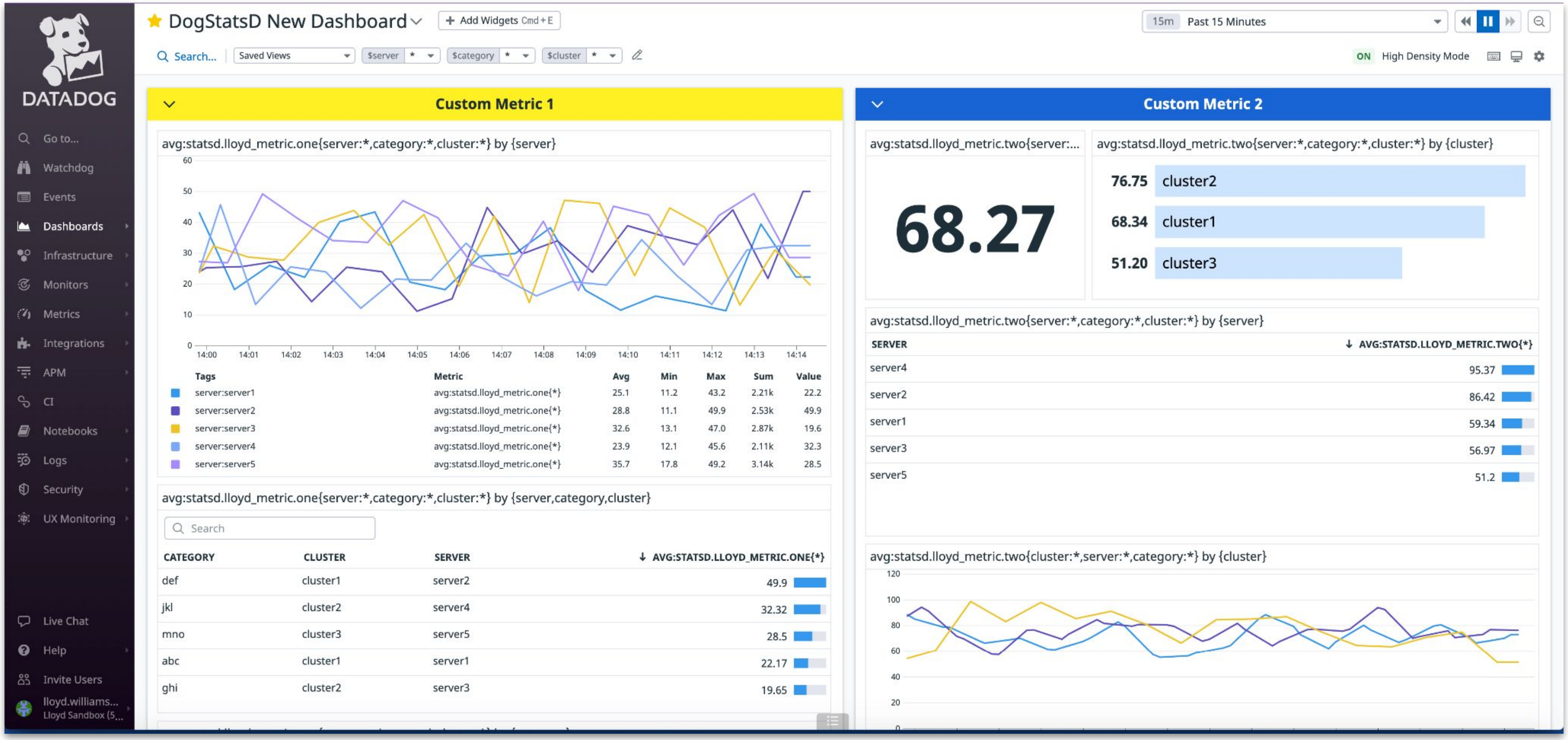
Export these graphs to: New dashboard | Existing dashboard

1h Jan 31, 1:01 pm – Jan 31, 2:01 pm

Want to apply advanced functions and more? Open in a notebook →

Datadog Custom Dashboards

Datadog Dashboard with Template Variables



Custom Metric 1

avg:statsd.lloyd_metric.one{server:*,category:*,cluster:*} by {server}

	Tags	Metric	Avg	Min	Max	Sum	Value
🟦	server:server1	avg:statsd.lloyd_metric.one{*}	25.1	11.2	43.2	2.21k	22.2
🟪	server:server2	avg:statsd.lloyd_metric.one{*}	28.8	11.1	49.9	2.53k	49.9
🟨	server:server3	avg:statsd.lloyd_metric.one{*}	32.6	13.1	47.0	2.87k	19.6
🟦	server:server4	avg:statsd.lloyd_metric.one{*}	23.9	12.1	45.6	2.11k	32.3
🟪	server:server5	avg:statsd.lloyd_metric.one{*}	35.7	17.8	49.2	3.14k	28.5

avg:statsd.lloyd_metric.one{server:*,category:*,cluster:*} by {server,category,cluster}

Search

CATEGORY	CLUSTER	SERVER	↓ AVG:STATSD.LLOYD_METRIC.ONE{*}
def	cluster1	server2	49.9
jkl	cluster2	server4	32.32
mno	cluster3	server5	28.5
abc	cluster1	server1	22.17
ghi	cluster2	server3	19.65

Custom Metric 2

avg:statsd.lloyd_metric.two{server:...

68.27

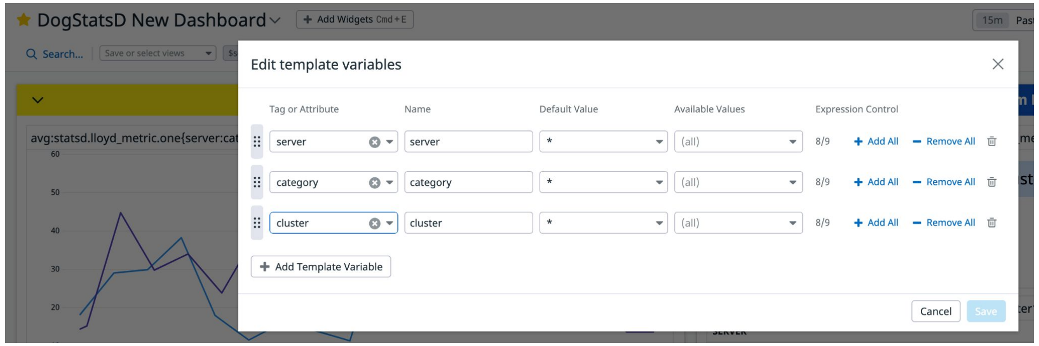
avg:statsd.lloyd_metric.two{server:*,category:*,cluster:*} by {cluster}

76.75	cluster2
68.34	cluster1
51.20	cluster3

avg:statsd.lloyd_metric.two{server:*,category:*,cluster:*} by {server}

SERVER	↓ AVG:STATSD.LLOYD_METRIC.TWO{*}
server4	95.37
server2	86.42
server1	59.34
server3	56.97
server5	51.2

avg:statsd.lloyd_metric.two{cluster:*,server:*,category:*} by {cluster}



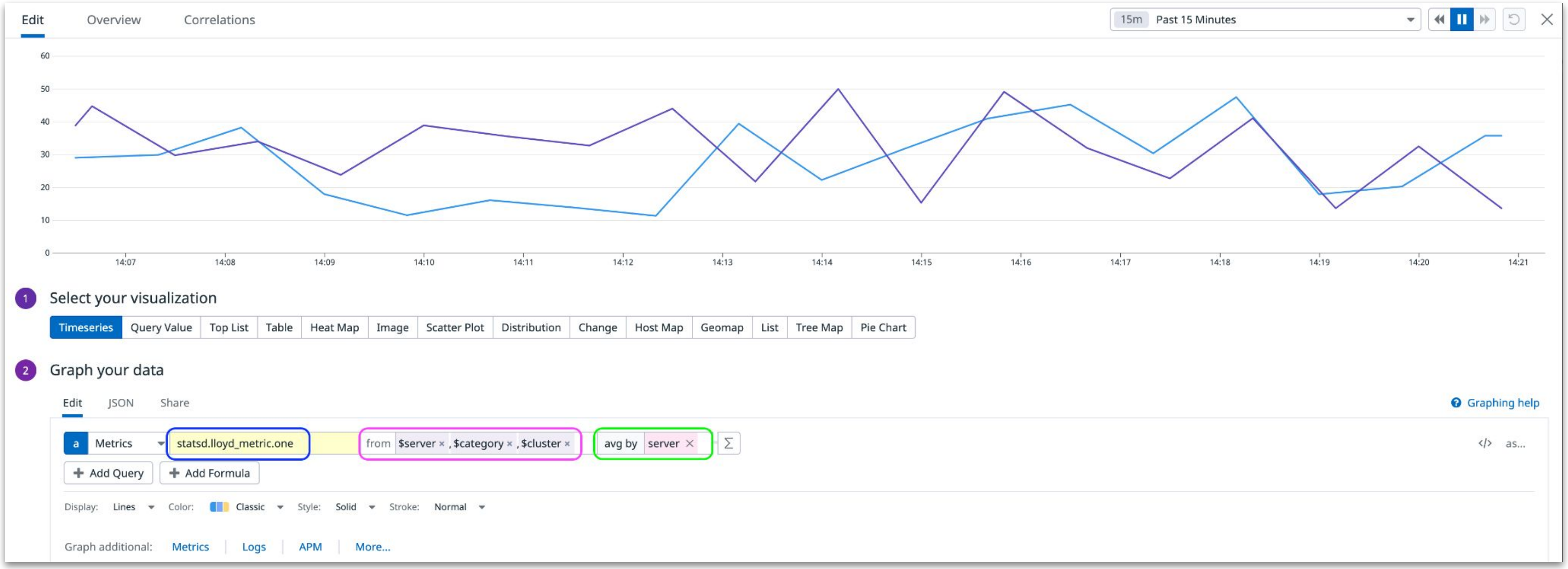
Edit template variables

Tag or Attribute	Name	Default Value	Available Values	Expression Control		
server	server	*	(all)	8/9	Add All	Remove All
category	category	*	(all)	8/9	Add All	Remove All
cluster	cluster	*	(all)	8/9	Add All	Remove All

You can set-up your dashboard with template variables to allow end-users to filter and analyze data from a single dashboard.

Datadog Custom Dashboards

Datadog Dashboard with Template Variables



1 Select your visualization

2 Graph your data

a Metrics statsd.lloyd_metric.one from $server × , $category × , $cluster × avg by server × Σ

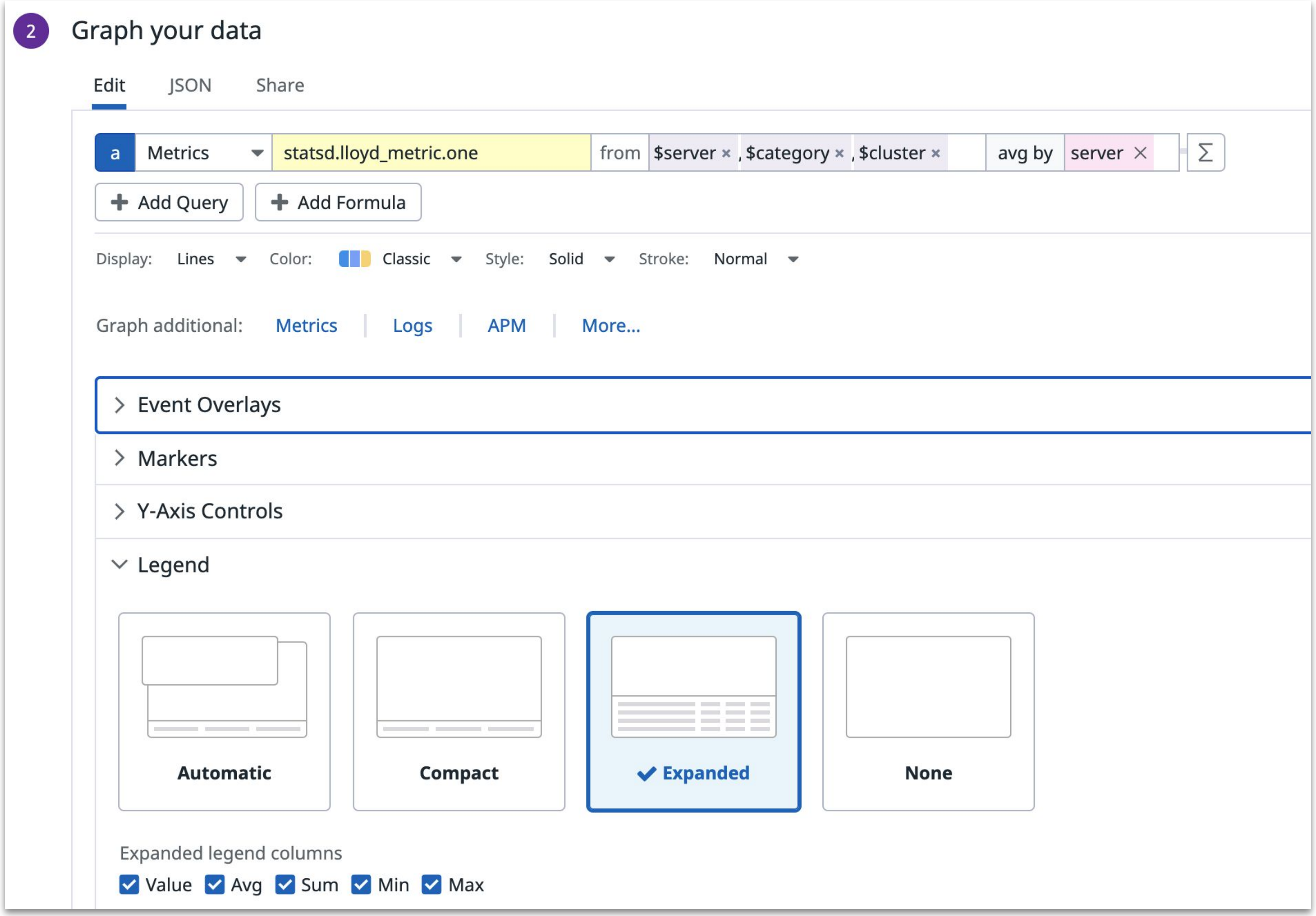
Display: Lines Color: Classic Style: Solid Stroke: Normal

Use '$' followed by the name of the template variable to filter the metric data by the template variables. Hover over the line to see the value at that particular time.

DATADOG

Datadog Custom Dashboards

Datadog Dashboard with Template Variables



2 Graph your data

Edit JSON Share

a Metrics statsd.lloyd_metric.one from $server × , $category × , $cluster × avg by server × Σ

Add Query Add Formula

Display: Lines Color: Classic Style: Solid Stroke: Normal

Graph additional: Metrics | Logs | APM | More...

Expanded legend columns

☑ Value ☑ Avg ☑ Sum ☑ Min ☑ Max

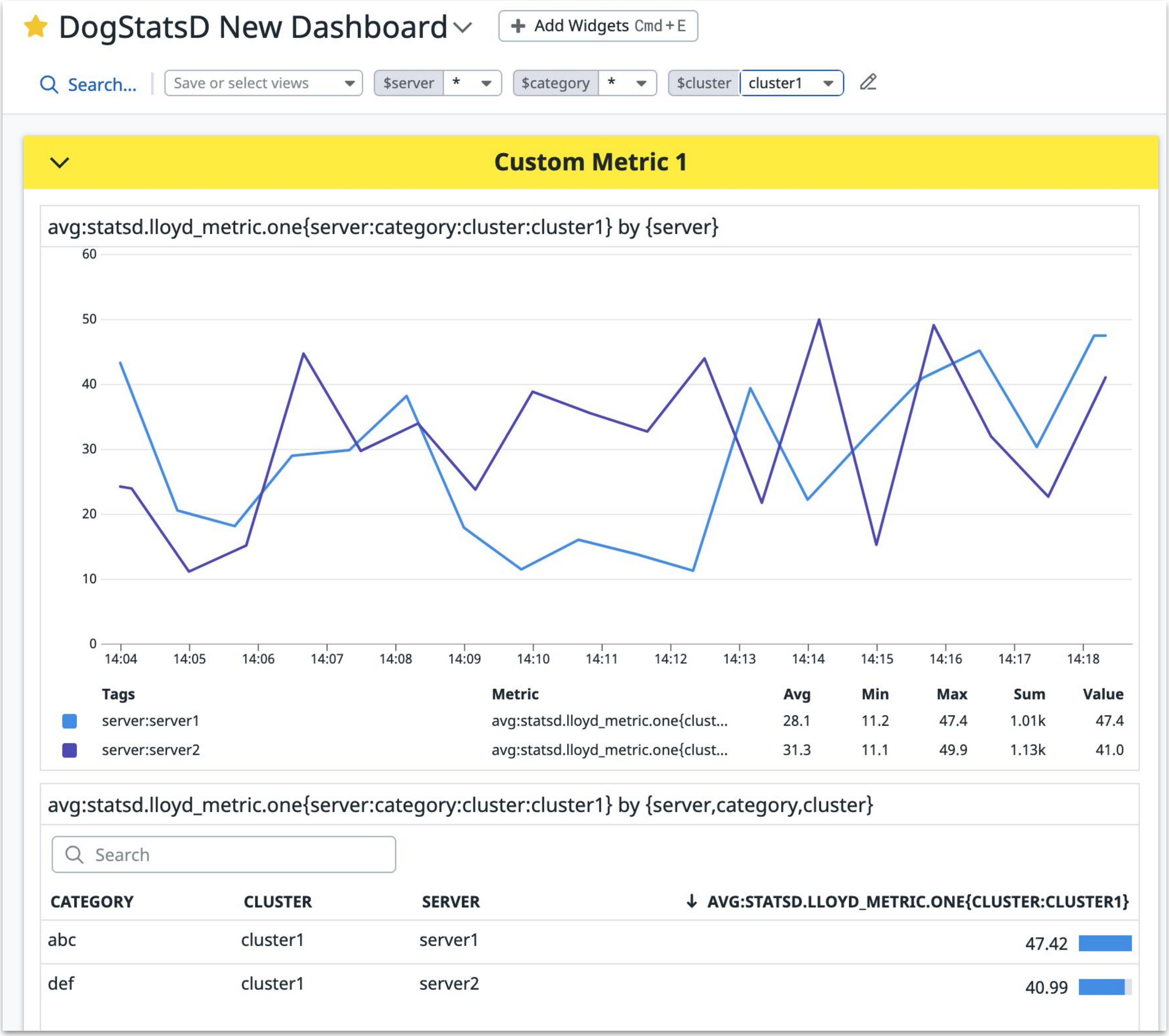
On a ‘timeseries’ graph, it can be useful to have a legend and have it display some additional calculated values.

DATADOG

Datadog Custom Dashboards

Datadog Dashboard with Template Variables

DATADOG



avg:statsd.lloyd_metric.one{server:category:cluster:cluster1} by {server}

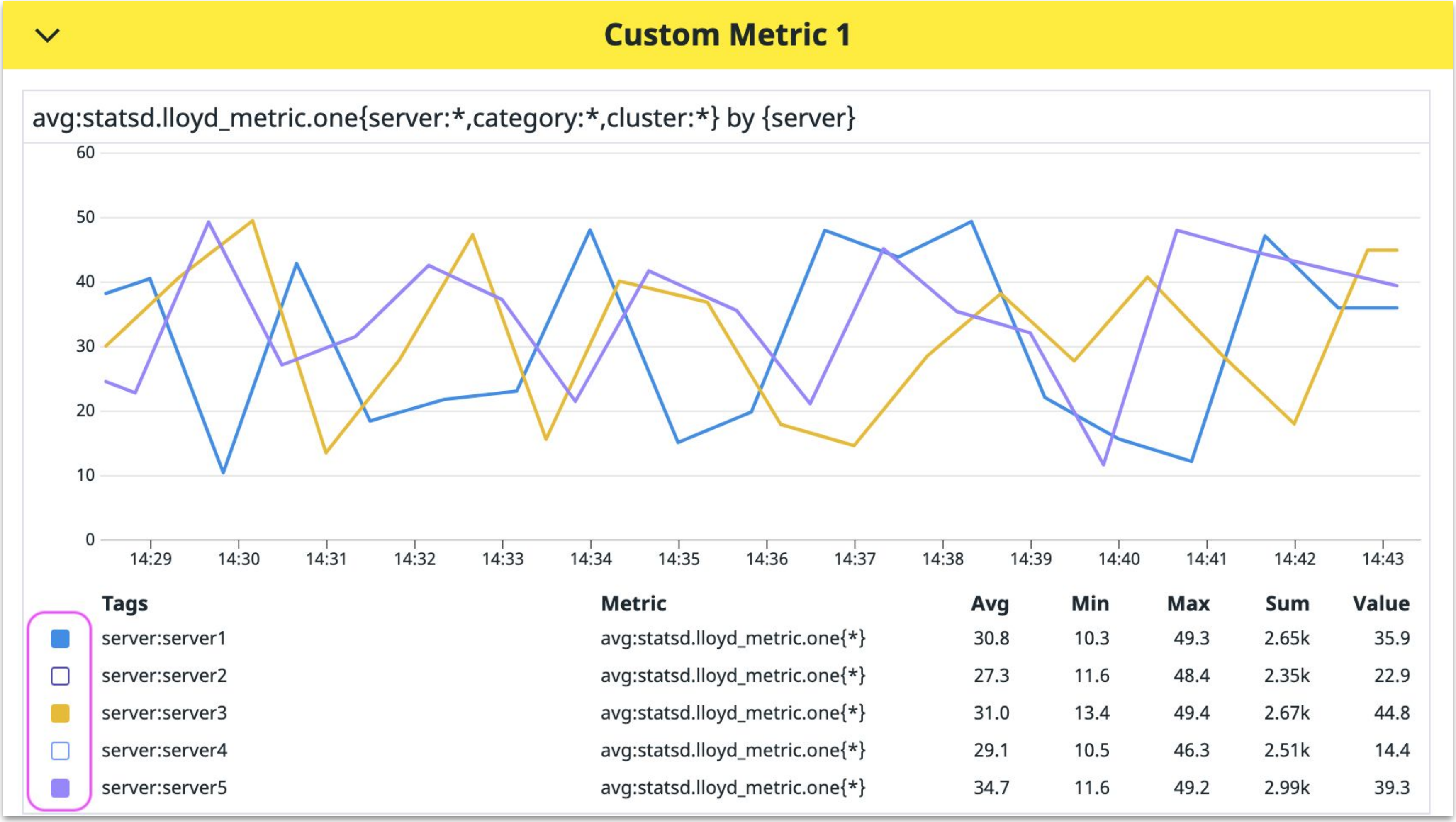
	Tags	Metric	Avg	Min	Max	Sum	Value
🟦	server:server1	avg:statsd.lloyd_metric.one{clust...	28.1	11.2	47.4	1.01k	47.4
🟪	server:server2	avg:statsd.lloyd_metric.one{clust...	31.3	11.1	49.9	1.13k	41.0

avg:statsd.lloyd_metric.one{server:category:cluster:cluster1} by {server,category,cluster}

CATEGORY	CLUSTER	SERVER	↓ AVG:STATSD.LLOYD_METRIC.ONE{CLUSTER:CLUSTER1}
abc	cluster1	server1	47.42
def	cluster1	server2	40.99

Datadog Custom Dashboards

Datadog Dashboard with Template Variables



Custom Metric 1

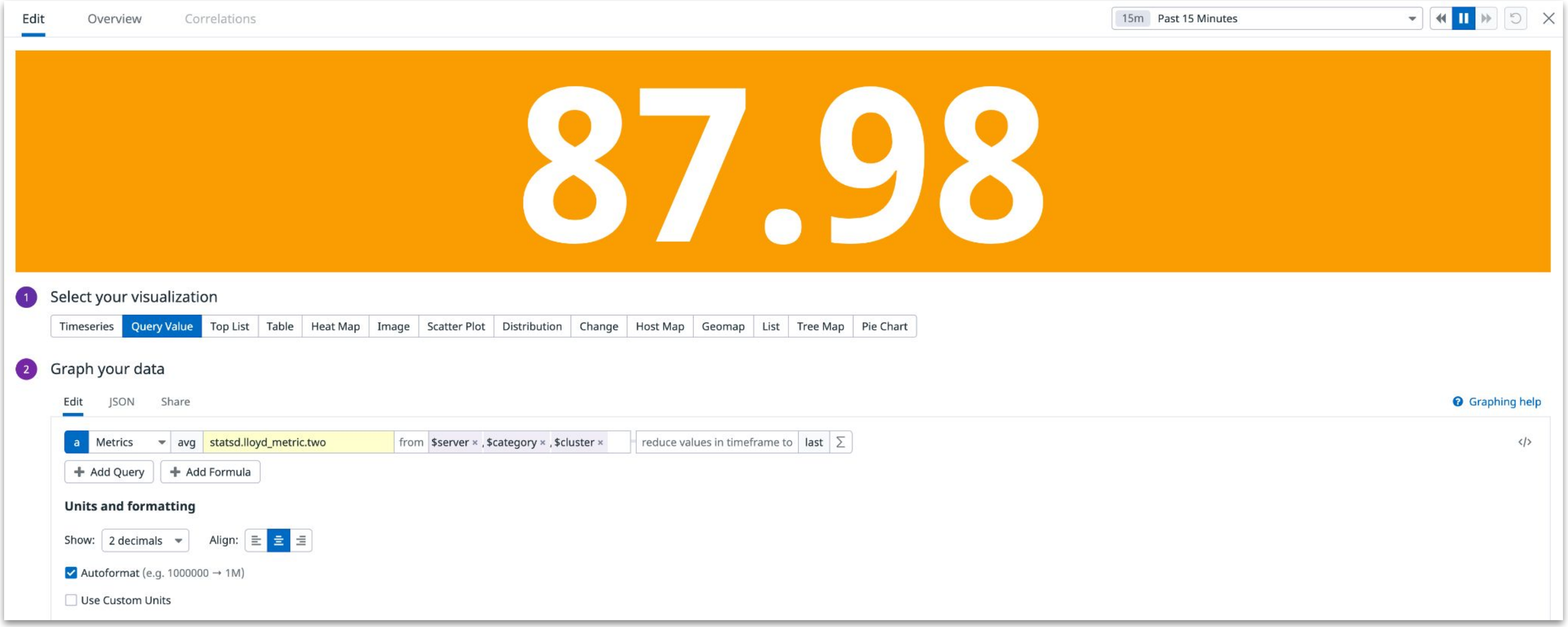
avg:statsd.lloyd_metric.one{server:*,category:*,cluster:*} by {server}

	Tags	Metric	Avg	Min	Max	Sum	Value
🟦	server:server1	avg:statsd.lloyd_metric.one{*}	30.8	10.3	49.3	2.65k	35.9
□	server:server2	avg:statsd.lloyd_metric.one{*}	27.3	11.6	48.4	2.35k	22.9
🟨	server:server3	avg:statsd.lloyd_metric.one{*}	31.0	13.4	49.4	2.67k	44.8
□	server:server4	avg:statsd.lloyd_metric.one{*}	29.1	10.5	46.3	2.51k	14.4
🟪	server:server5	avg:statsd.lloyd_metric.one{*}	34.7	11.6	49.2	2.99k	39.3

You'll also notice that you can select and unselect the boxes from the legend which gives the user greater control over what they want to view.

DATADOG

Datadog Custom Dashboards



15m Past 15 Minutes

87.98

1 Select your visualization: Query Value

2 Graph your data

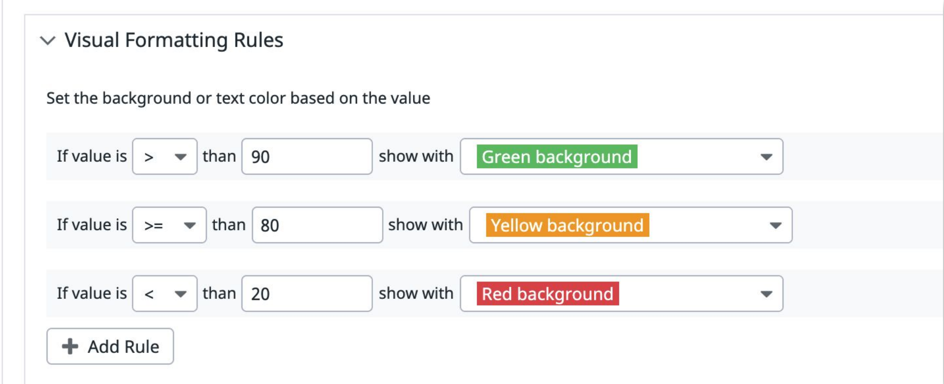
a Metrics avg statsd.lloyd_metric.two from $server × , $category × , $cluster × reduce values in timeframe to last Σ

Units and formatting

Show: 2 decimals Align:

☑ Autoformat (e.g. 1000000 → 1M)

□ Use Custom Units



Visual Formatting Rules

Set the background or text color based on the value

If value is > than 90 show with Green background

If value is >= than 80 show with Yellow background

If value is < than 20 show with Red background

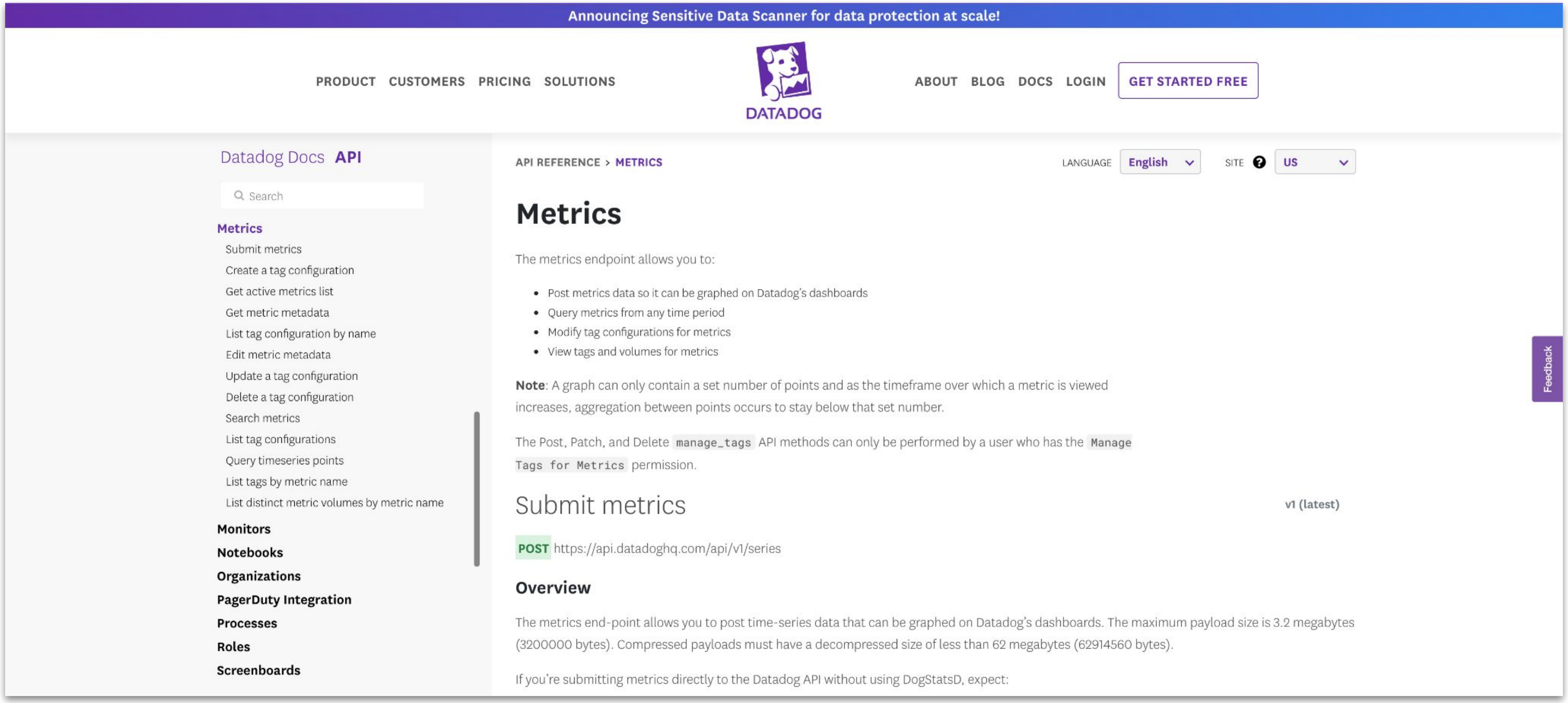
You can set a ‘Query Value’ visualization that simply shows the latest value, by using ‘reduce values in timeframe to’ = ‘last’ and apply some rules to set the background color.

DATADOG

Custom Metrics via API

Datadog REST APIs

Datadog APIs allow you to perform specific tasks programmatically like sending custom metrics.



API REFERENCE > METRICS

LANGUAGE English

SITE US

Metrics

The metrics endpoint allows you to:

- Post metrics data so it can be graphed on Datadog's dashboards
- Query metrics from any time period
- Modify tag configurations for metrics
- View tags and volumes for metrics

Note: A graph can only contain a set number of points and as the timeframe over which a metric is viewed increases, aggregation between points occurs to stay below that set number.

The Post, Patch, and Delete `manage_tags` API methods can only be performed by a user who has the `Manage Tags for Metrics` permission.

Submit metrics

v1 (latest)

POST https://api.datadoghq.com/api/v1/series

Overview

The metrics end-point allows you to post time-series data that can be graphed on Datadog's dashboards. The maximum payload size is 3.2 megabytes (3200000 bytes). Compressed payloads must have a decompressed size of less than 62 megabytes (62914560 bytes).

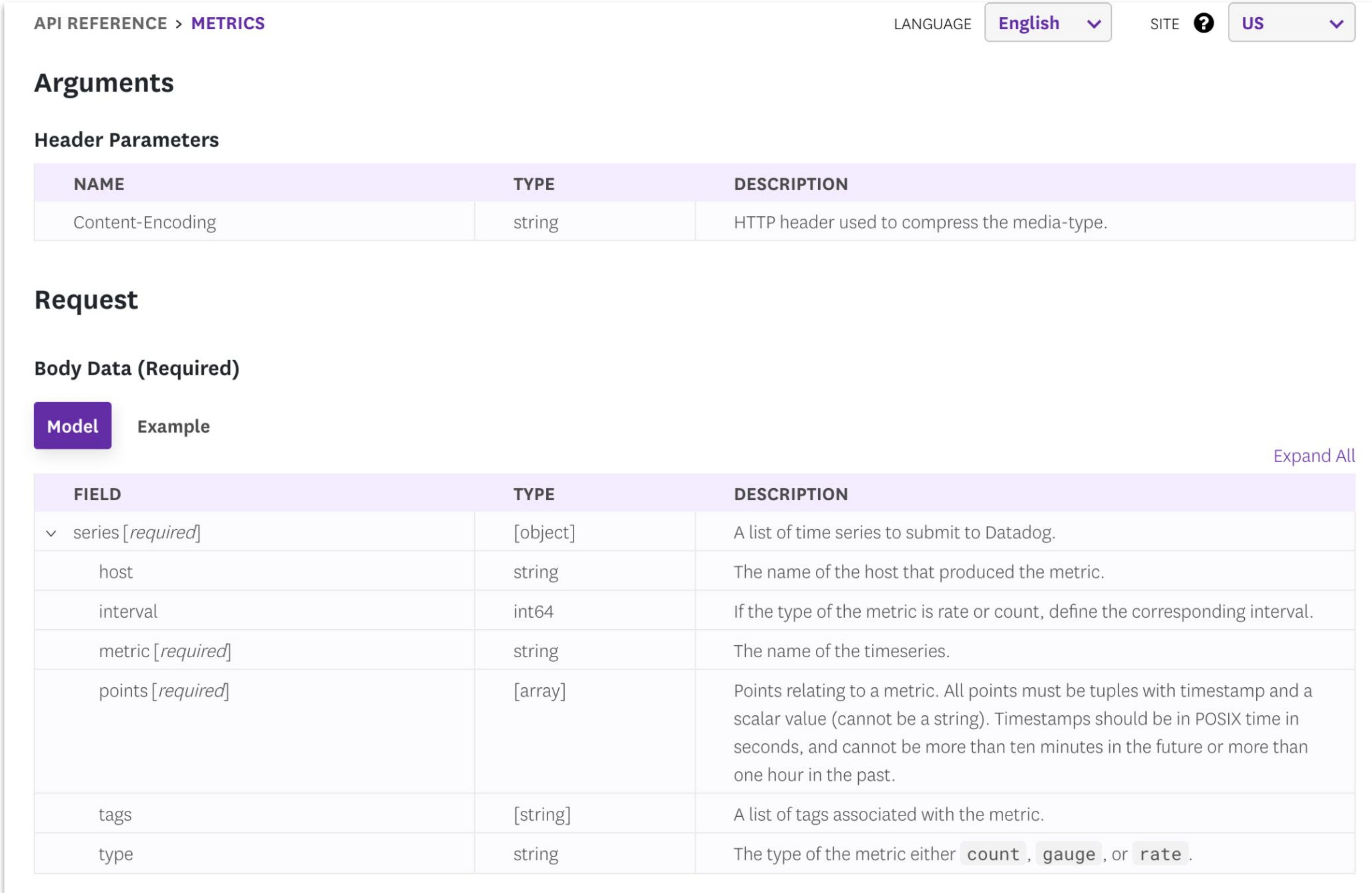
If you're submitting metrics directly to the Datadog API without using DogStatsD, expect:

Customers can use the Datadog API to send custom metrics. The Datadog online documentation explains the format of the body of the message and provides an example.

Datadog REST APIs

Datadog APIs allow you to perform specific tasks programmatically like sending custom metrics.

DATADOG



LANGUAGE English

SITE US

Arguments

Header Parameters

NAME	TYPE	DESCRIPTION
Content-Encoding	string	HTTP header used to compress the media-type.

Request

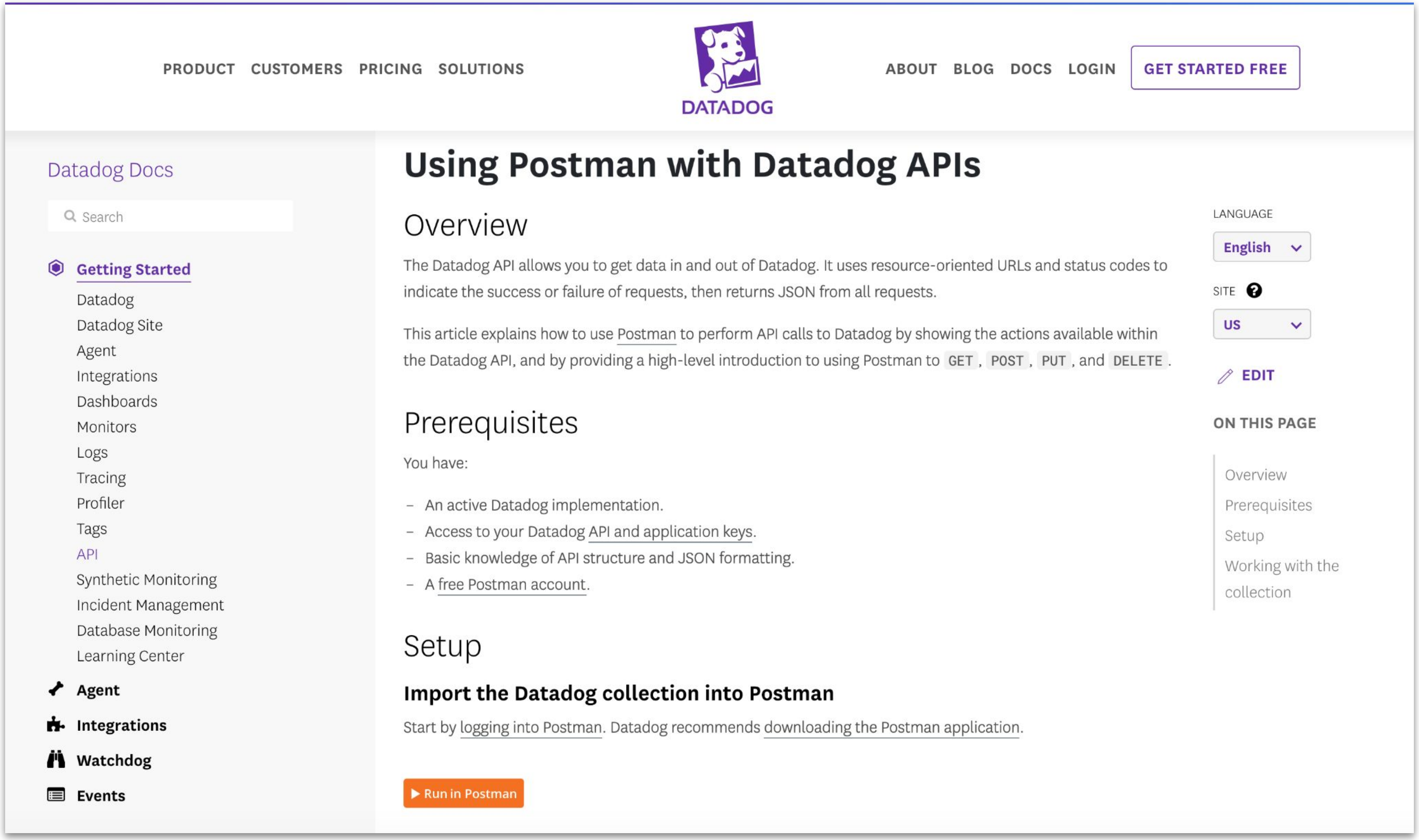
Body Data (Required)

FIELD	TYPE	DESCRIPTION
series [*required*]	[object]	A list of time series to submit to Datadog.
host	string	The name of the host that produced the metric.
interval	int64	If the type of the metric is rate or count, define the corresponding interval.
metric [*required*]	string	The name of the timeseries.
points [*required*]	[array]	Points relating to a metric. All points must be tuples with timestamp and a scalar value (cannot be a string). Timestamps should be in POSIX time in seconds, and cannot be more than ten minutes in the future or more than one hour in the past.
tags	[string]	A list of tags associated with the metric.
type	string	The type of the metric either `count` , `gauge` , or `rate` .

Customers can use the Datadog API to send custom metrics. The Datadog online documentation explains the format of the body of the message and provides an example.

Datadog REST APIs

Datadog provides an integration to Postman to allow you to see examples of the API calls.



Using Postman with Datadog APIs

Overview

The Datadog API allows you to get data in and out of Datadog. It uses resource-oriented URLs and status codes to indicate the success or failure of requests, then returns JSON from all requests.

This article explains how to use Postman to perform API calls to Datadog by showing the actions available within the Datadog API, and by providing a high-level introduction to using Postman to `GET`, `POST`, `PUT`, and `DELETE`.

Prerequisites

You have:

- An active Datadog implementation.
- Access to your Datadog API and application keys.
- Basic knowledge of API structure and JSON formatting.
- A free Postman account.

Setup

Import the Datadog collection into Postman

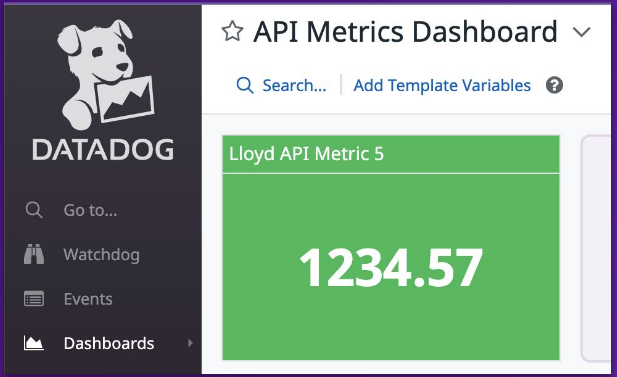
Start by logging into Postman. Datadog recommends downloading the Postman application.

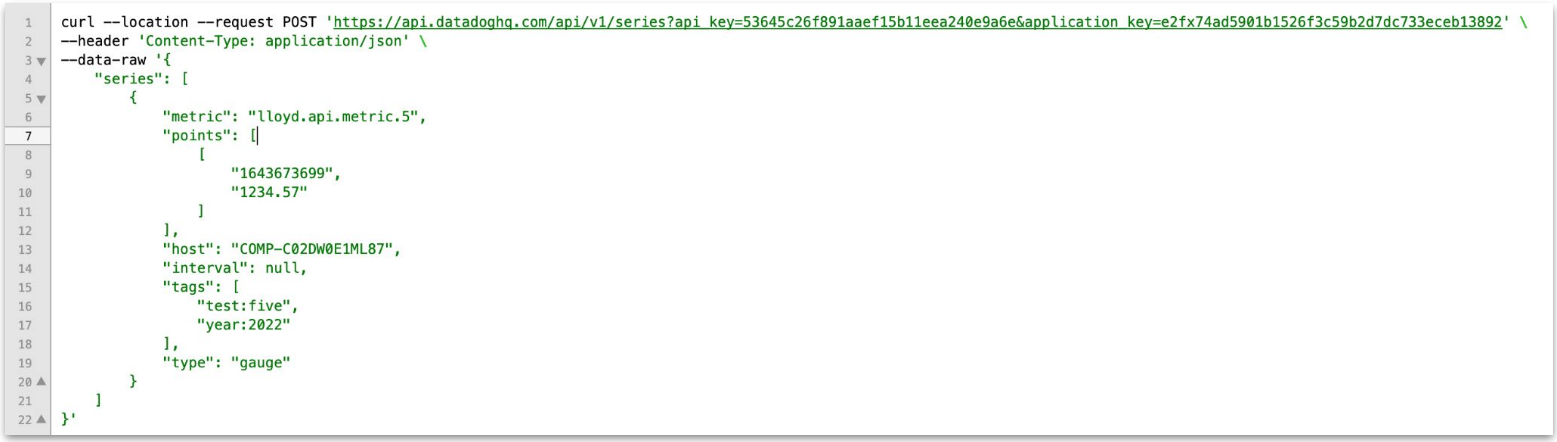
In order to send the metrics data directly to Datadog SaaS, you must specify valid **API and Application key** which can be generated in the “Organization Settings” of the Datadog SaaS UI.

DATADOG

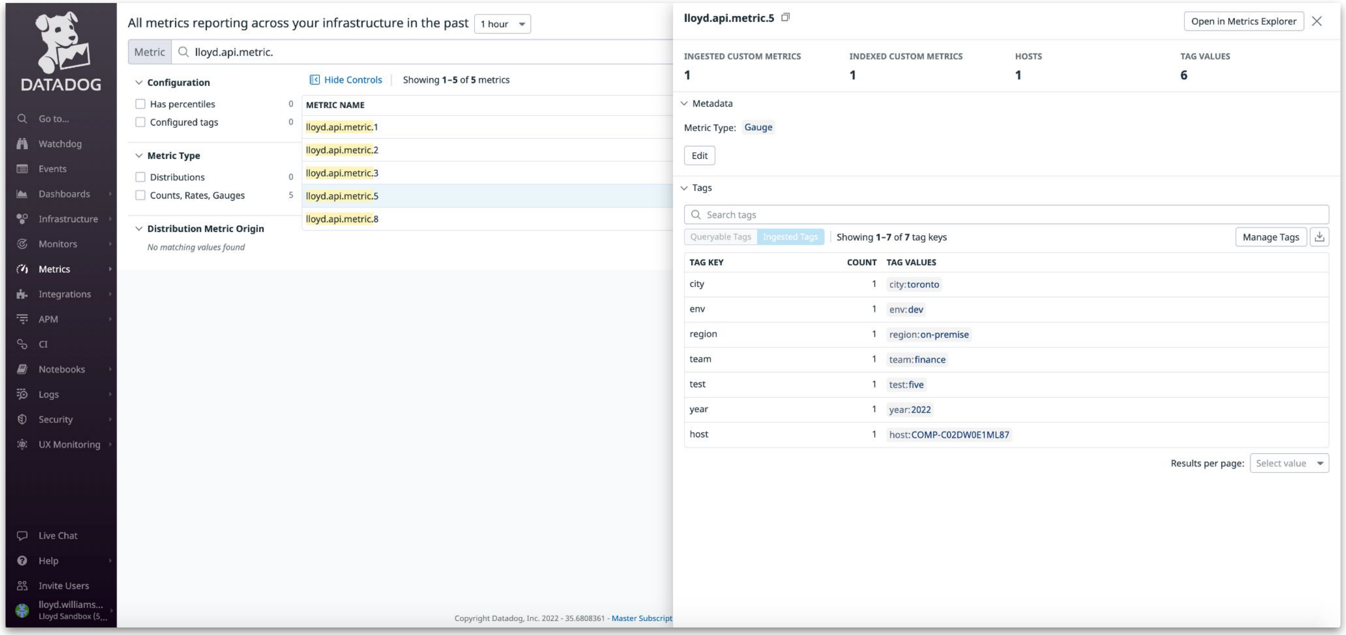
Datadog REST APIs

POST metrics example





```
curl --location --request POST 'https://api.datadoghq.com/api/v1/series?api_key=53645c26f891aaef15b11eea240e9a6e&application_key=e2fx74ad5901b1526f3c59b2d7dc733eceb13892' \
--header 'Content-Type: application/json' \
--data-raw '{
    "series": [
        {
            "metric": "lloyd.api.metric.5",
            "points": [
                [
                    "1643673699",
                    "1234.57"
                ]
            ],
            "host": "COMP-C02DW0E1ML87",
            "interval": null,
            "tags": [
                "test:five",
                "year:2022"
            ],
            "type": "gauge"
        }
    ]
}'
```



All metrics reporting across your infrastructure in the past 1 hour

Metric: lloyd.api.metric.

Configuration

☐ Has percentiles 0
☐ Configured tags 0

Metric Type

☐ Distributions 0
☐ Counts, Rates, Gauges 5

Distribution Metric Origin

No matching values found

Showing **1–5** of **5** metrics

METRIC NAME
lloyd.api.metric.1
lloyd.api.metric.2
lloyd.api.metric.3
lloyd.api.metric.5
lloyd.api.metric.8

lloyd.api.metric.5

INGESTED CUSTOM METRICS	INDEXED CUSTOM METRICS	HOSTS	TAG VALUES
1	1	1	6

Metadata

Metric Type: Gauge

Tags

Showing **1–7** of **7** tag keys

TAG KEY	COUNT	TAG VALUES
city	1	city:toronto
env	1	env:dev
region	1	region:on-premise
team	1	team:finance
test	1	test:five
year	1	year:2022
host	1	host:COMP-C02DW0E1ML87

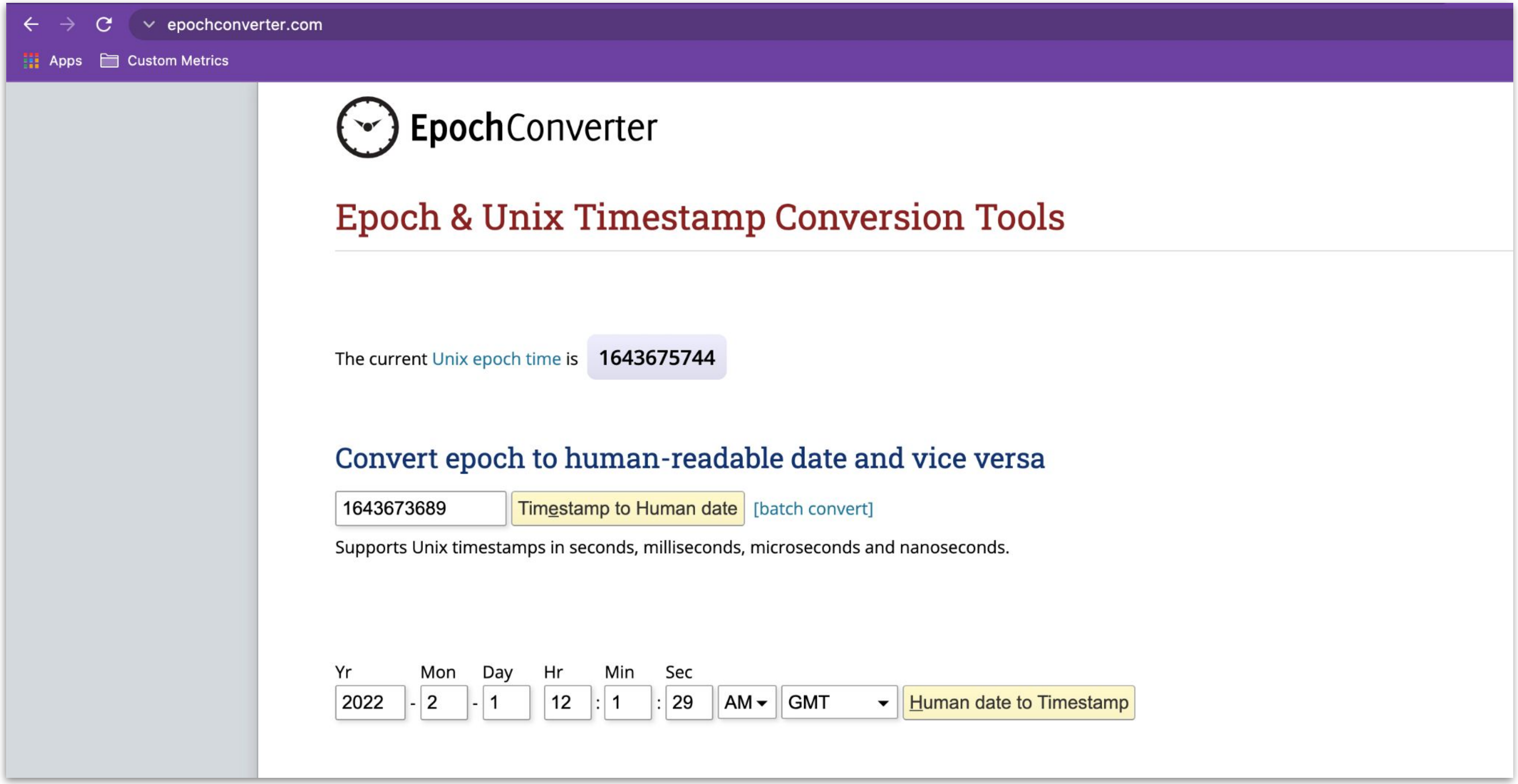
Results per page: Select value

In order to send the metrics data directly to Datadog SaaS, you must specify valid **API and Application key** which can be generated in the “Organization Settings” of the Datadog SaaS UI.

DATADOG

Datadog REST APIs

Date / Time Conversion



Epoch & Unix Timestamp Conversion Tools

The current Unix epoch time is 1643675744

Convert epoch to human-readable date and vice versa

1643673689 Timestamp to Human date [batch convert]

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Yr		Mon		Day	Hr		Min		Sec			
2022	-	2	-	1	12	:	1	:	29	AM	GMT	Human date to Timestamp

The timestamp is Unix epoch time.

DATADOG

DATADOG