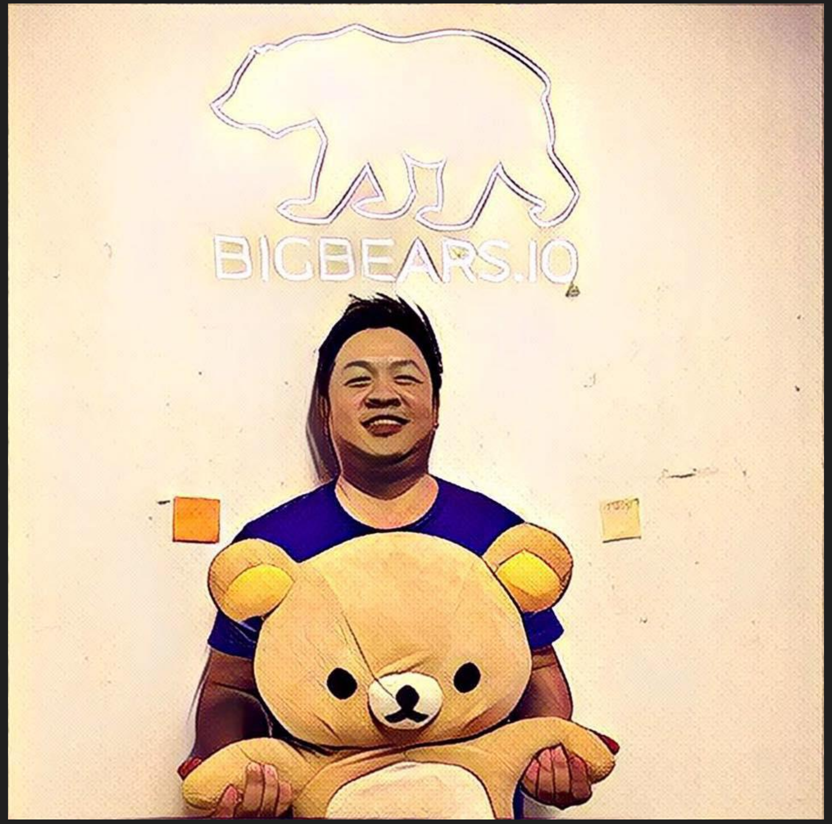
BIGBEARS.IO

GIVING ASP.NET WEBSITE VISIBILITY WITH REALTIME METRICS

MAHASAK PIJITTUM

BEARCHITECT



MAHASAK PIJITTUM

- BEARCHITECT @ BigBears.IO
- Senior Software Engineer @ Agoda
- Chief Party Officer @ ตุ่นนินจา
- พ่อบ้านใจกล้า @ Home
- ป๋า...

BIGBEARS.IO

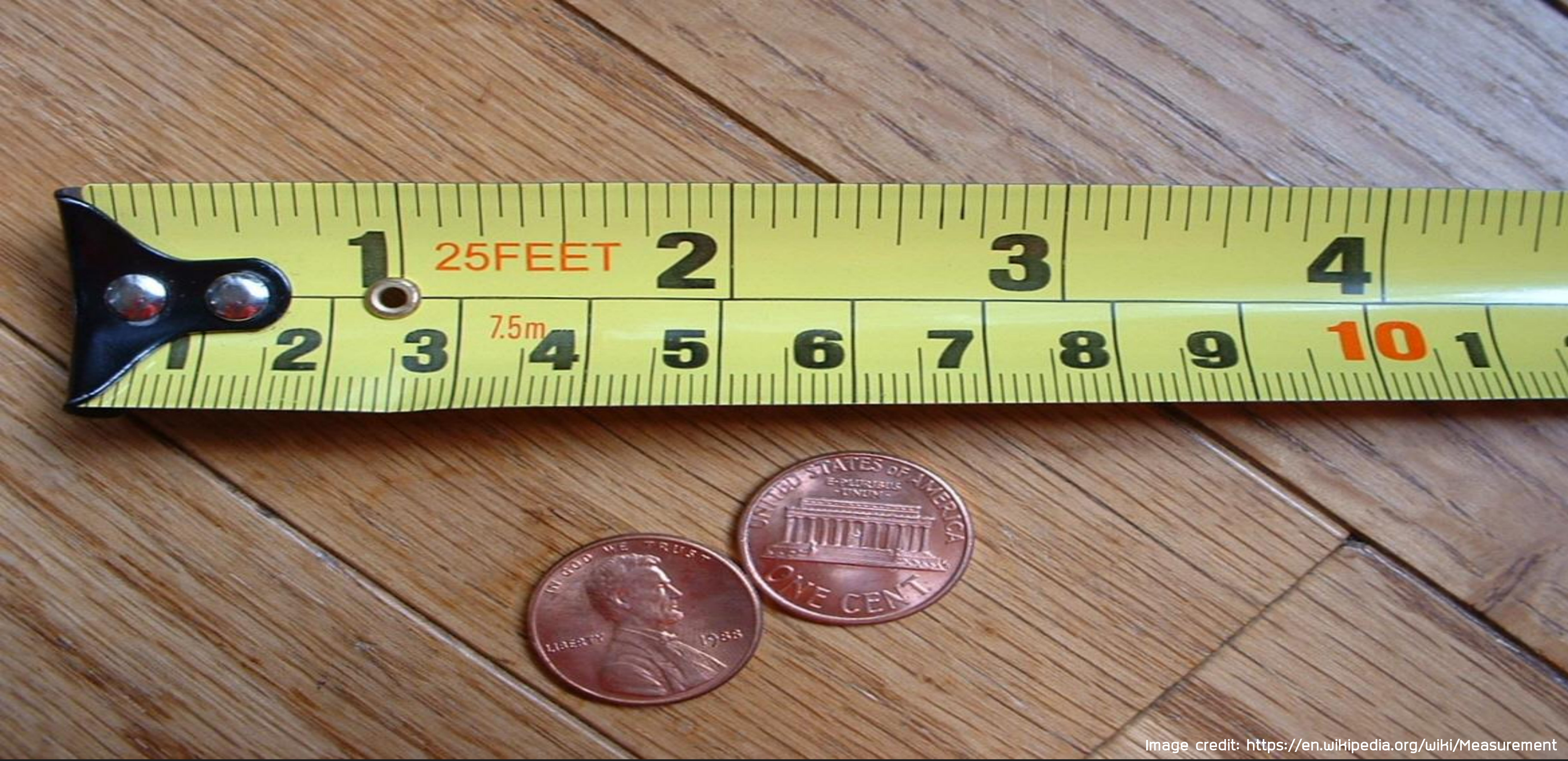


Image credit: https://en.wikipedia.org/wiki/Measurement

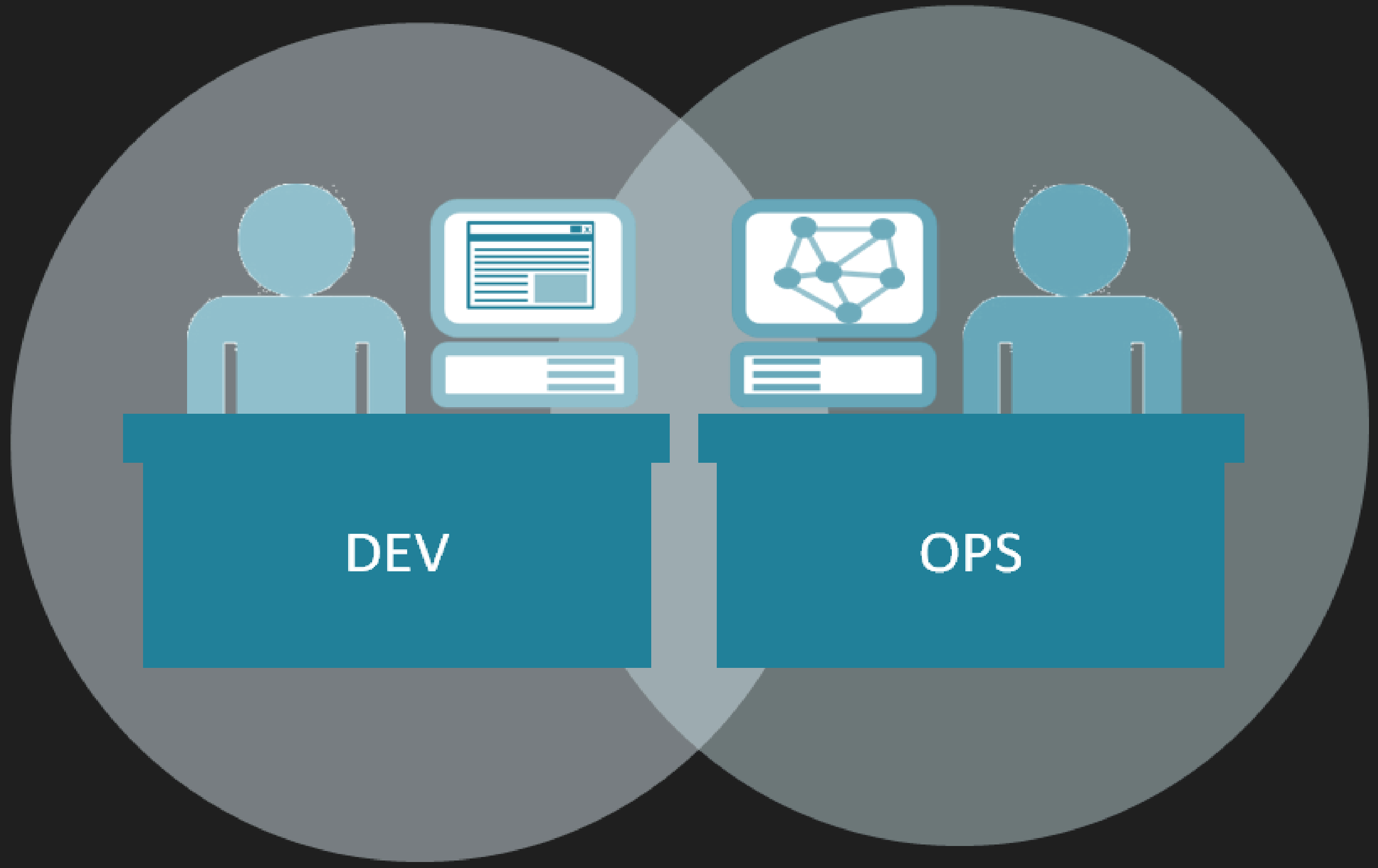
BIGBEARS.IO

Why Measurement

Why Measurement

TRACK SUCCESS	Track your growth and identify your challenges
IMPROVE	You can't improve what you can't measure
SET EXPECTATIONS	Where your expectation are in next 3 months
HELP DECISION	You can't decide without information

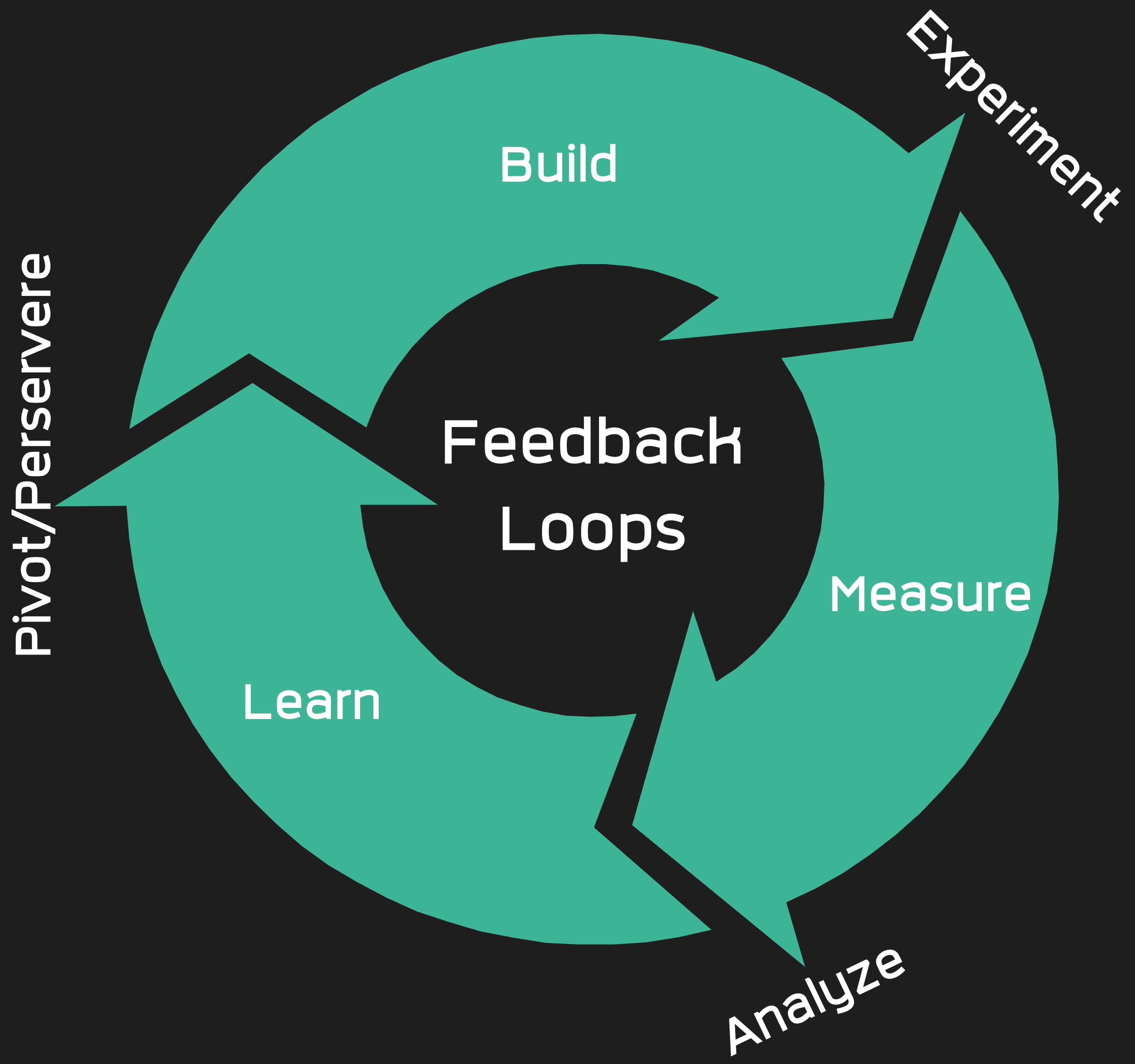
BIGBEARS.IO



BIGBEARS.IO

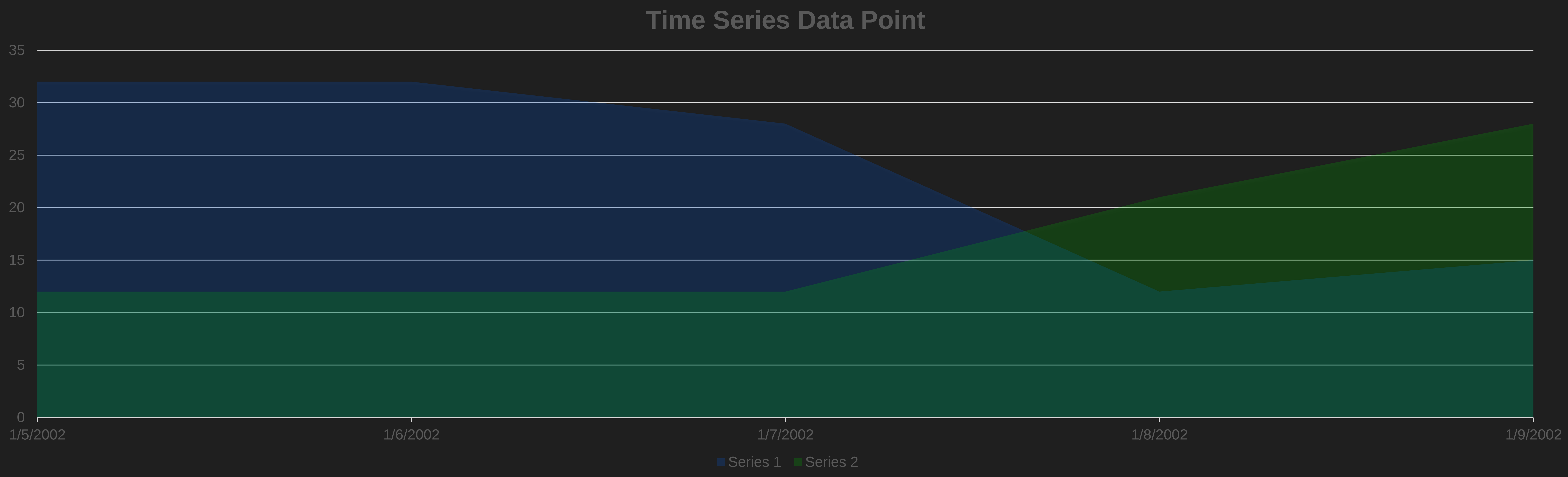
Who Loves Measurement

Data-Driven Strategy Concepts



BIGBEARS.IO

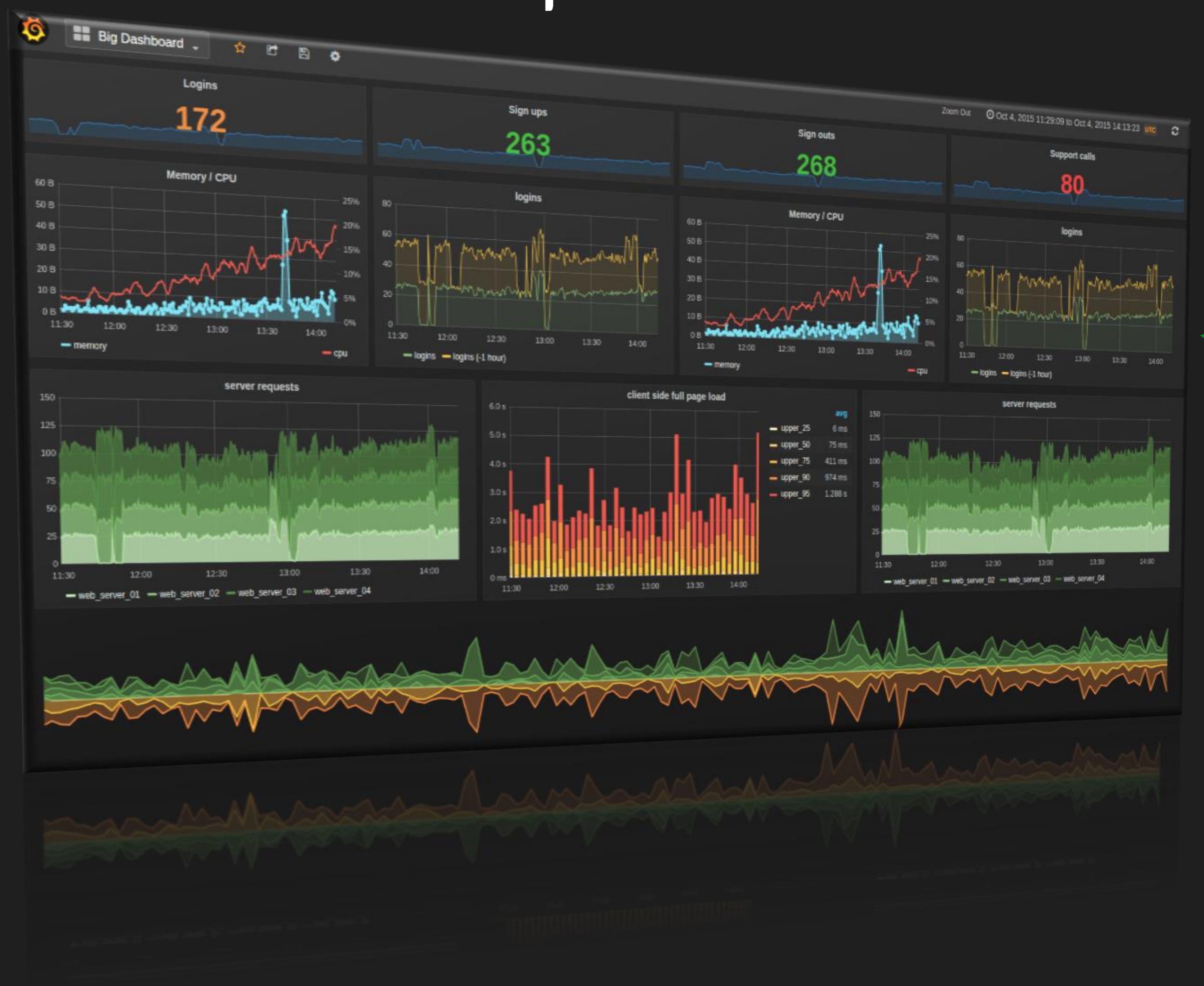
Information that represent "data point" for each point in time



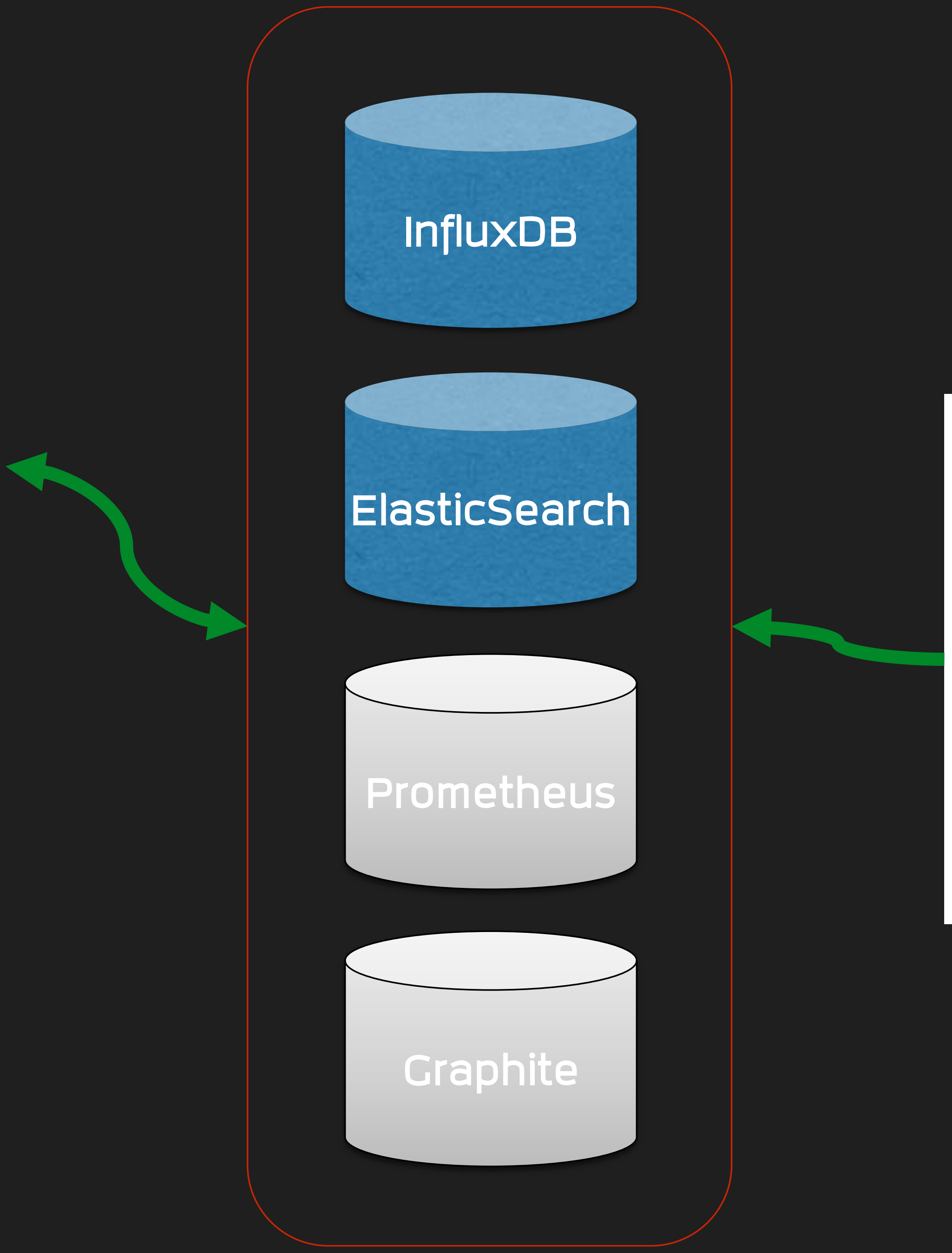
BIGBEARS.IO

Time Series Database

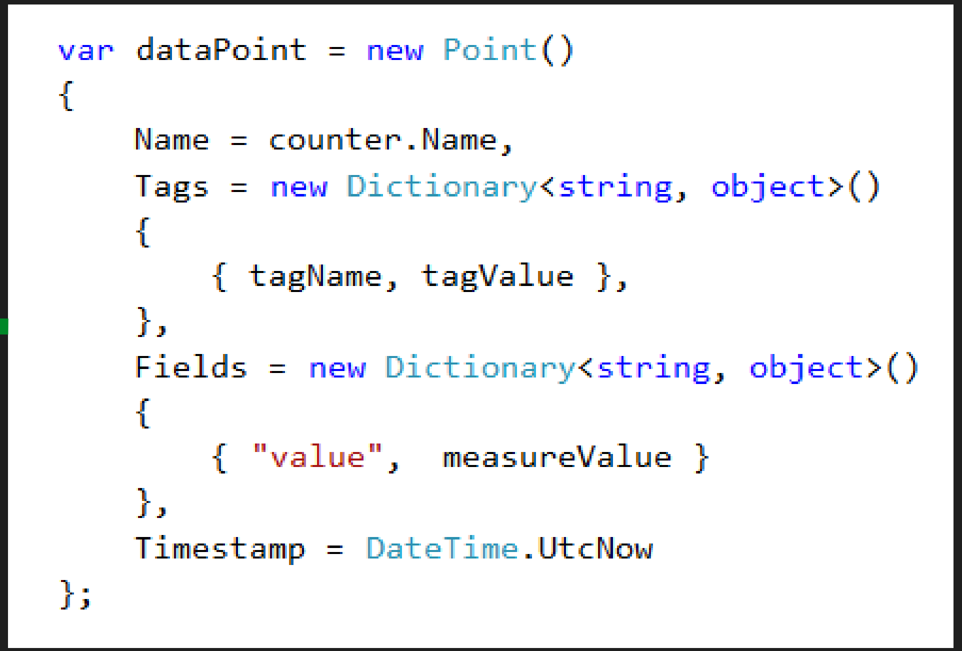
Grafana



Time Series Database



Metrics/Data Point



```
var dataPoint = new Point()
{
    Name = counter.Name,
    Tags = new Dictionary<string, object>()
    {
        { tagName, tagValue },
    },
    Fields = new Dictionary<string, object>()
    {
        { "value",  measureValue }
    },
    Timestamp = DateTime.UtcNow
};
```

BIGBEARS.IO

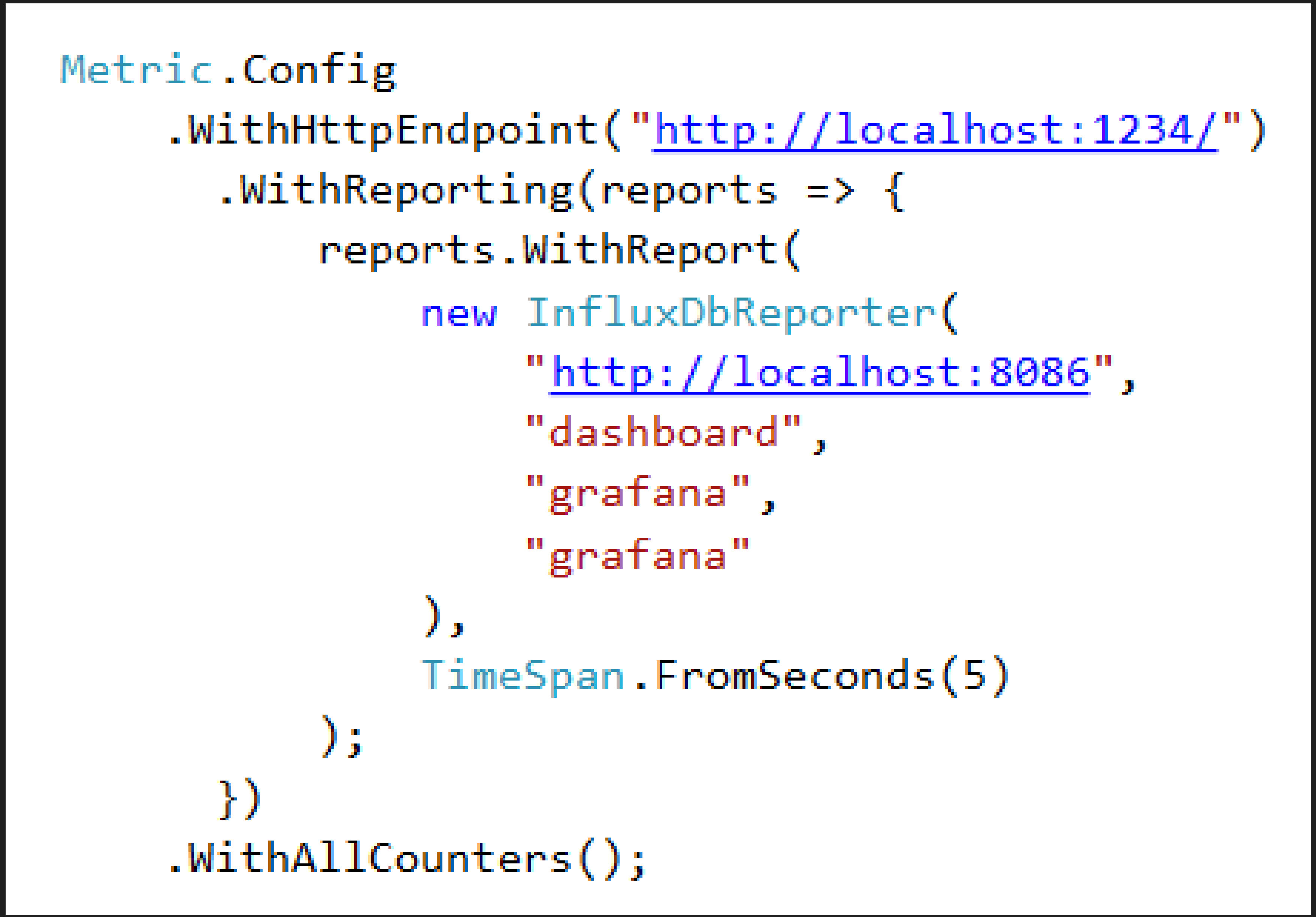
What you need

NuGet Packages

Metrics.Net	https://github.com/Recognos/Metrics.NET Install-Package Metrics.NET
InfluxData.Net	https://github.com/pootzko/InfluxData.Net Install-Package InfluxData.Net

BIGBEARS.IO

What you need



```
Metric.Config
    .WithHttpEndpoint("http://localhost:1234/")
      .WithReporting(reports => {
          reports.WithReport(
              new InfluxDbReporter(
                  "http://localhost:8086",
                  "dashboard",
                  "grafana",
                  "grafana"
              ),
              TimeSpan.FromSeconds(5)
          );
      })
    .WithAllCounters();
```

<- Visualization (Not for production)

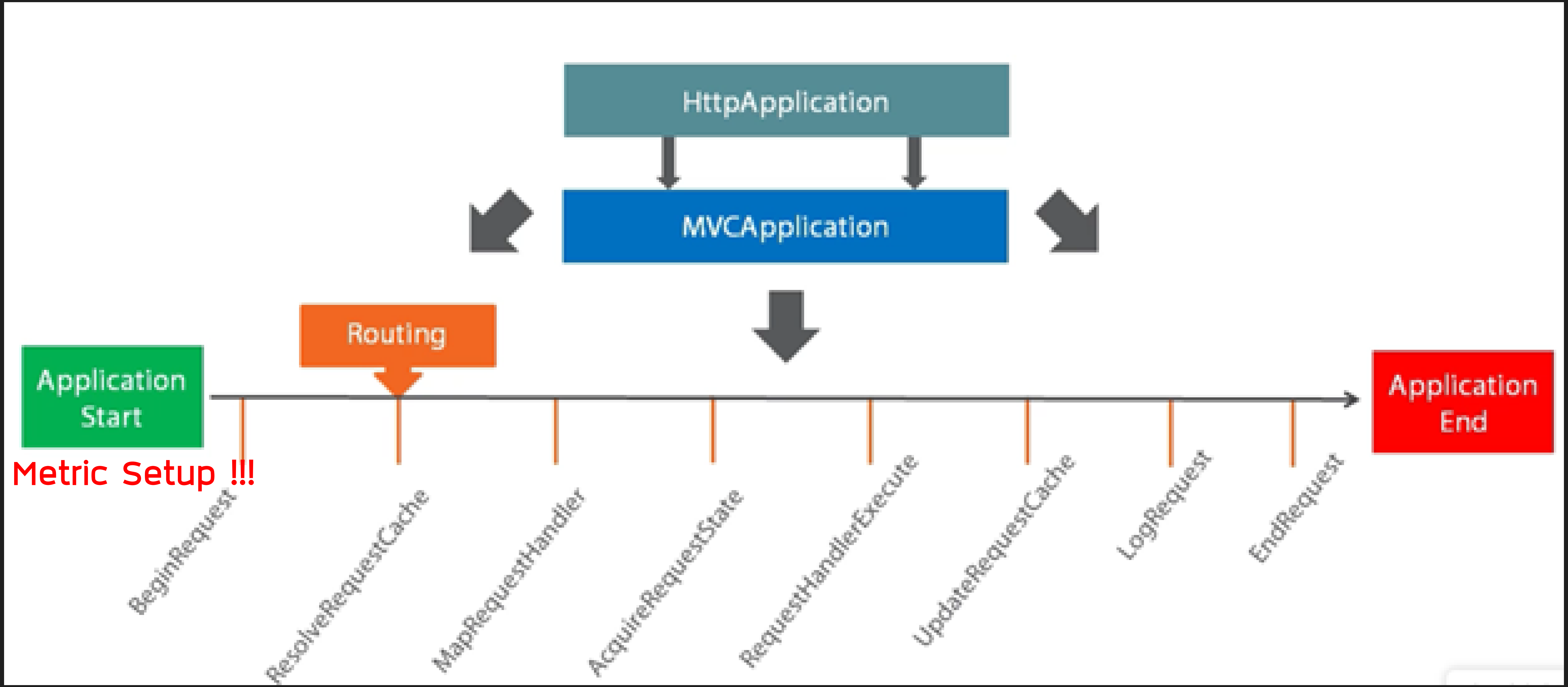
<- Data Point Reporter

<- Reporting Interval

<- Measurement(s) to report

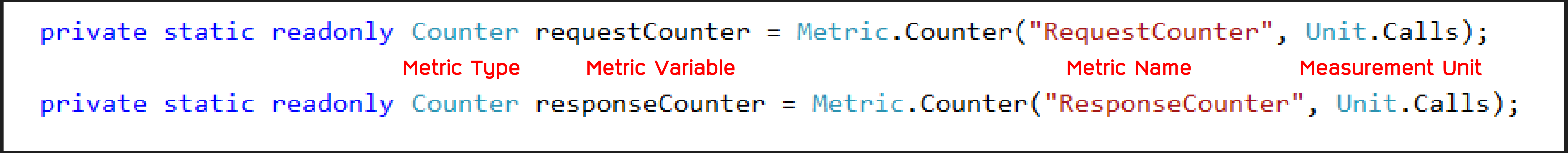
BIGBEARS.IO

How to Measure



BIGBEARS.IO

How to Measure



```
private static readonly Counter requestCounter = Metric.Counter("RequestCounter", Unit.Calls);
                        Metric Type  Metric Variable                 Metric Name         Measurement Unit
private static readonly Counter responseCounter = Metric.Counter("ResponseCounter", Unit.Calls);
```

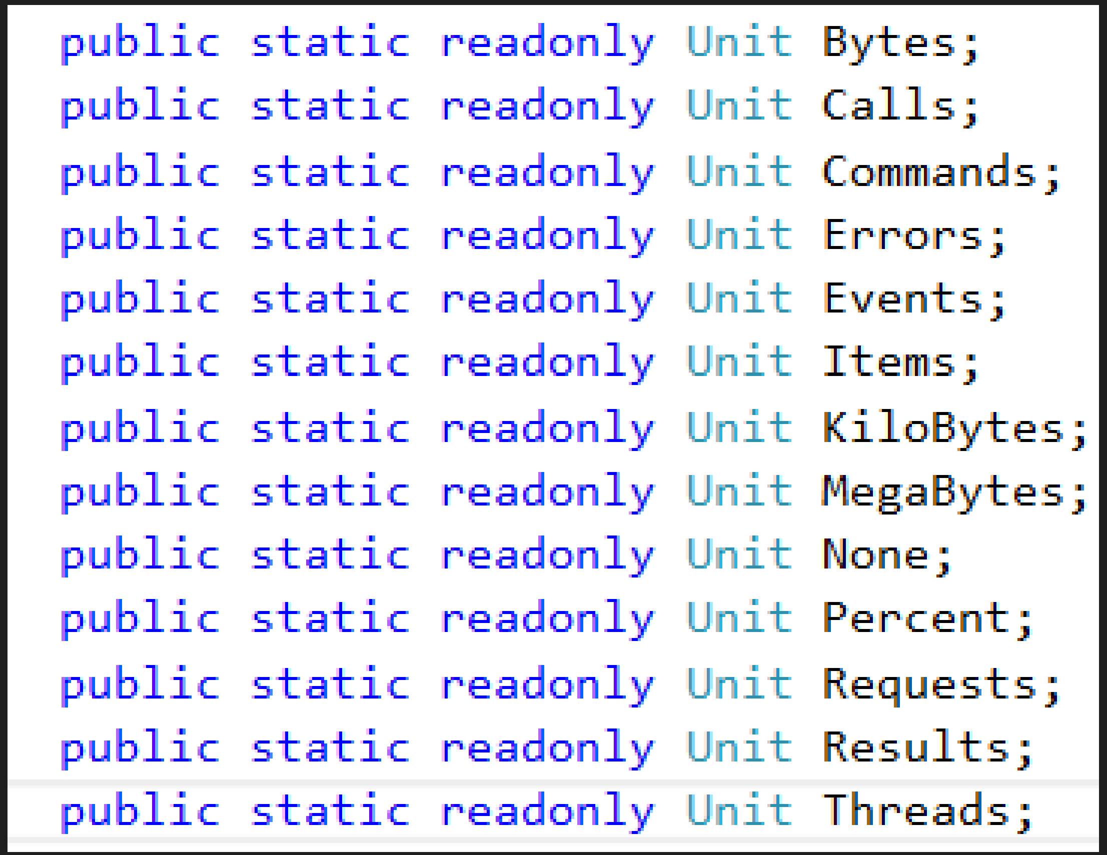
COUNTER 64-bit integer values that can be incremented and decremented

METER measures the rate at which an event occurs. (Counter, Mean, 1/5/15 minute rates)

HISTOGRAM measures the distribution of values and calculate Stats(Mean, Std.Dev, Median, Percentiles)

GUAGE represents an instantaneous value

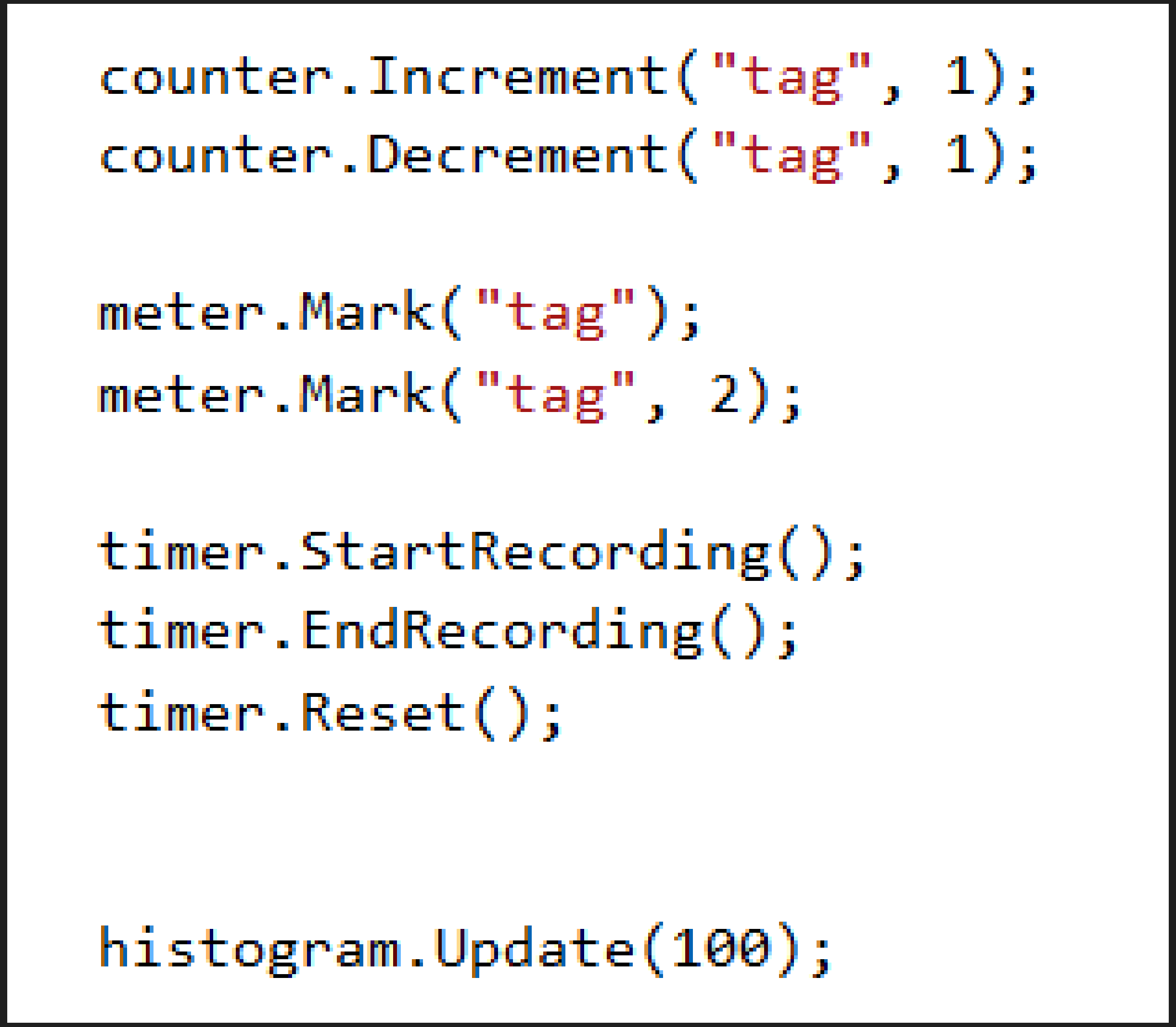
TIMER a histogram of the duration of a type of event and a meter of the rate of its occurrence



```
public static readonly Unit Bytes;
public static readonly Unit Calls;
public static readonly Unit Commands;
public static readonly Unit Errors;
public static readonly Unit Events;
public static readonly Unit Items;
public static readonly Unit KiloBytes;
public static readonly Unit MegaBytes;
public static readonly Unit None;
public static readonly Unit Percent;
public static readonly Unit Requests;
public static readonly Unit Results;
public static readonly Unit Threads;
```

BIGBEARS.IO

How to Measure



```
counter.Increment("tag", 1);
counter.Decrement("tag", 1);

meter.Mark("tag");
meter.Mark("tag", 2);

timer.StartRecording();
timer.EndRecording();
timer.Reset();


histogram.Update(100);
```

<- Counter

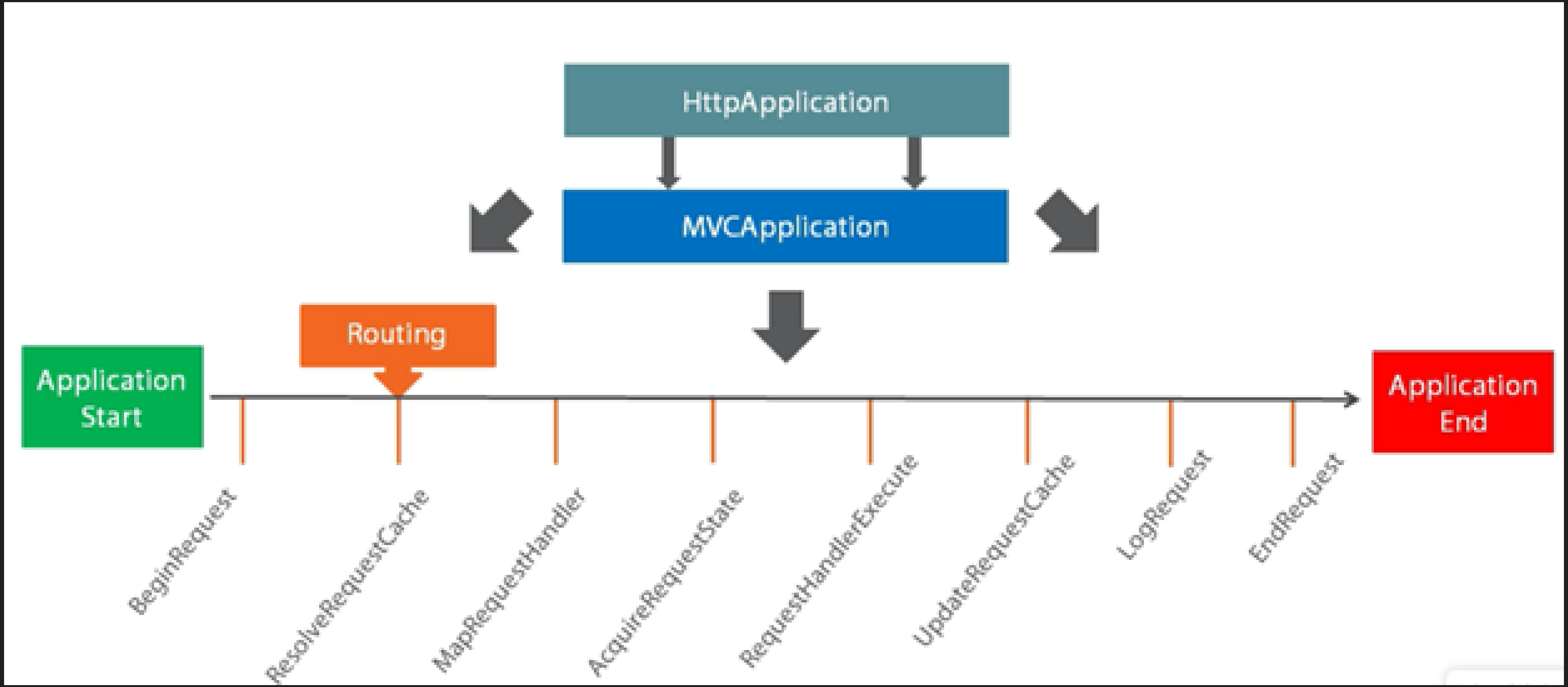
<- Meter

<- Timer

<- Histogram

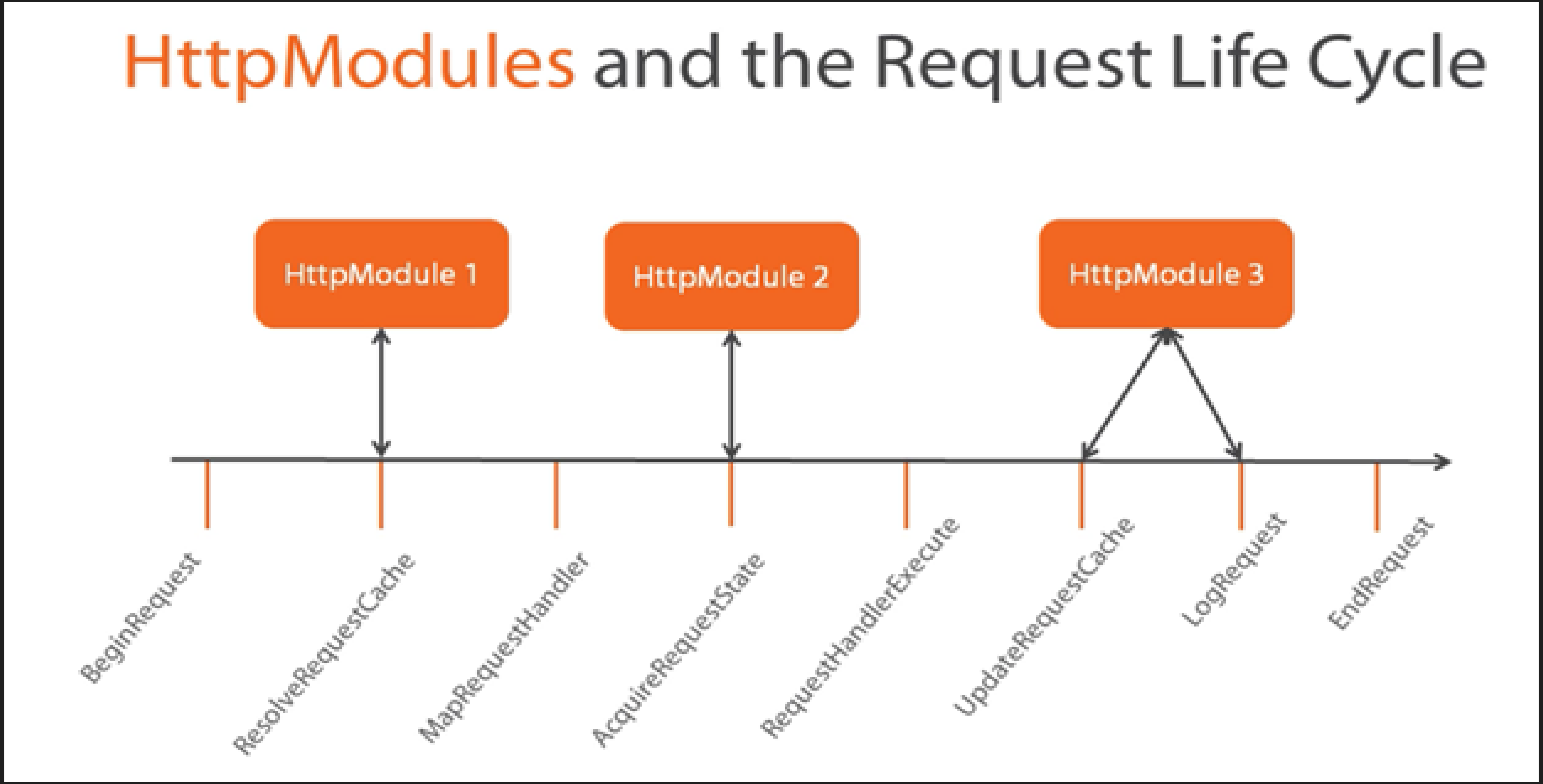
How to Measure

BIGBEARS.IO



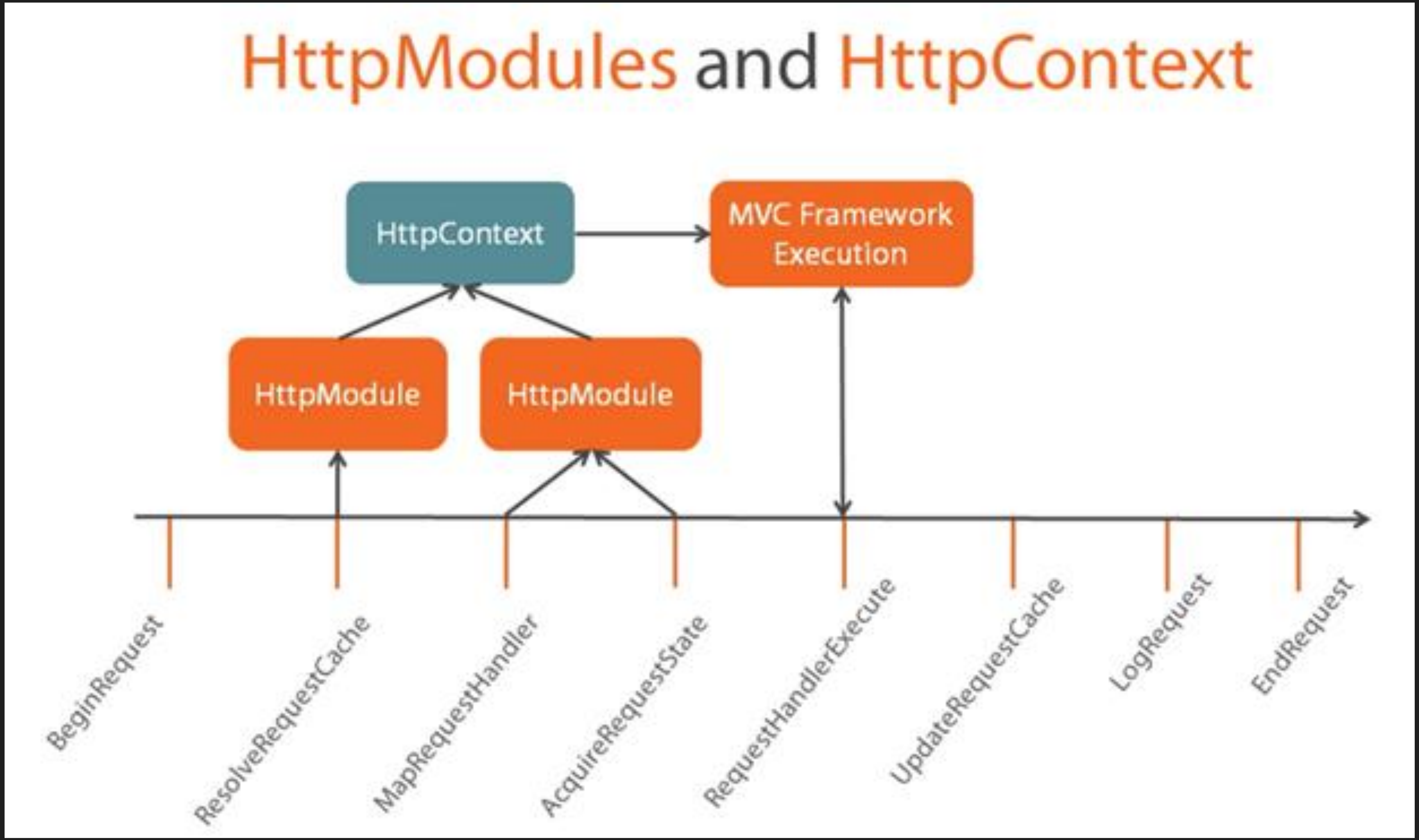
BIGBEARS.IO

Where to measure



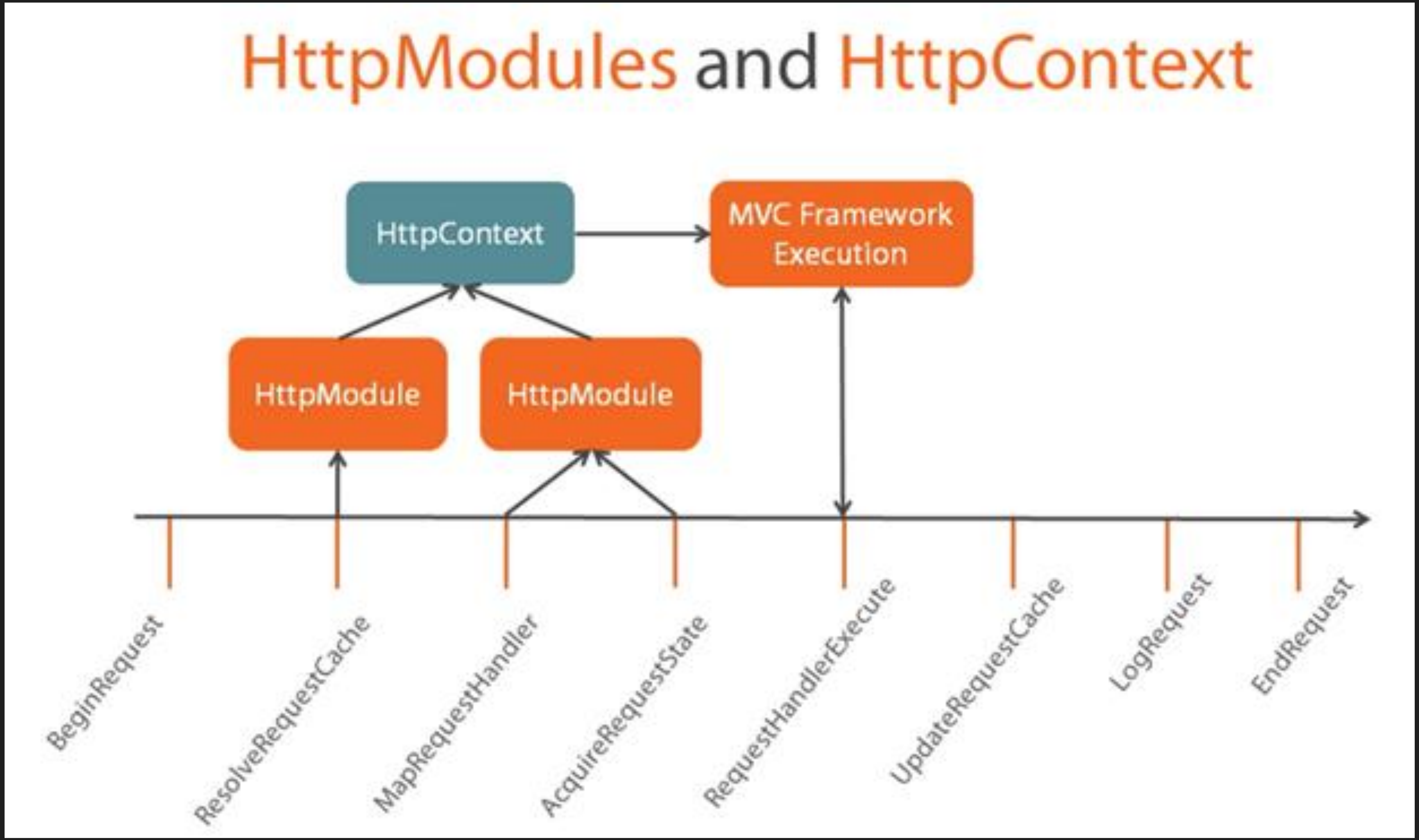
BIGBEARS.IO

Where to measure



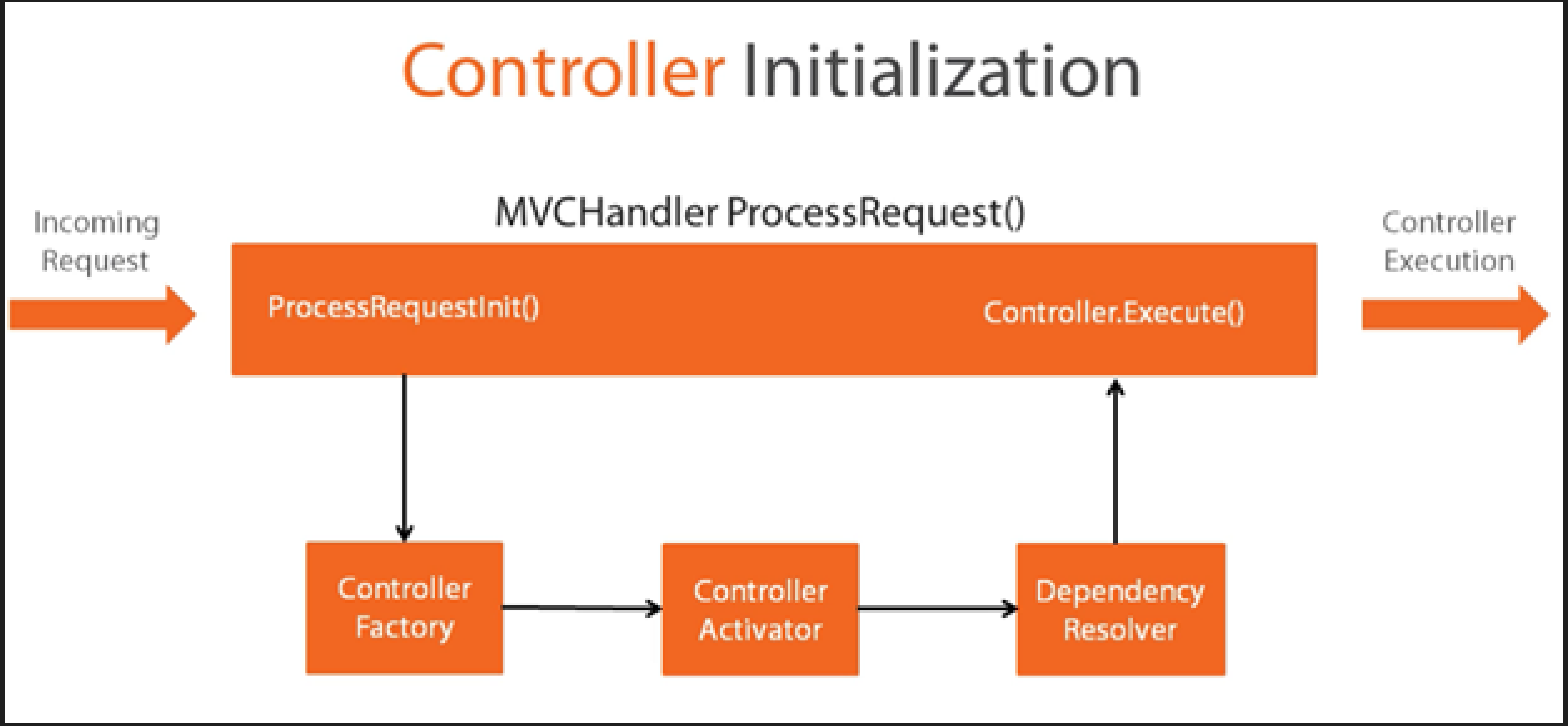
Where to measure

BIGBEARS.IO



Where to measure

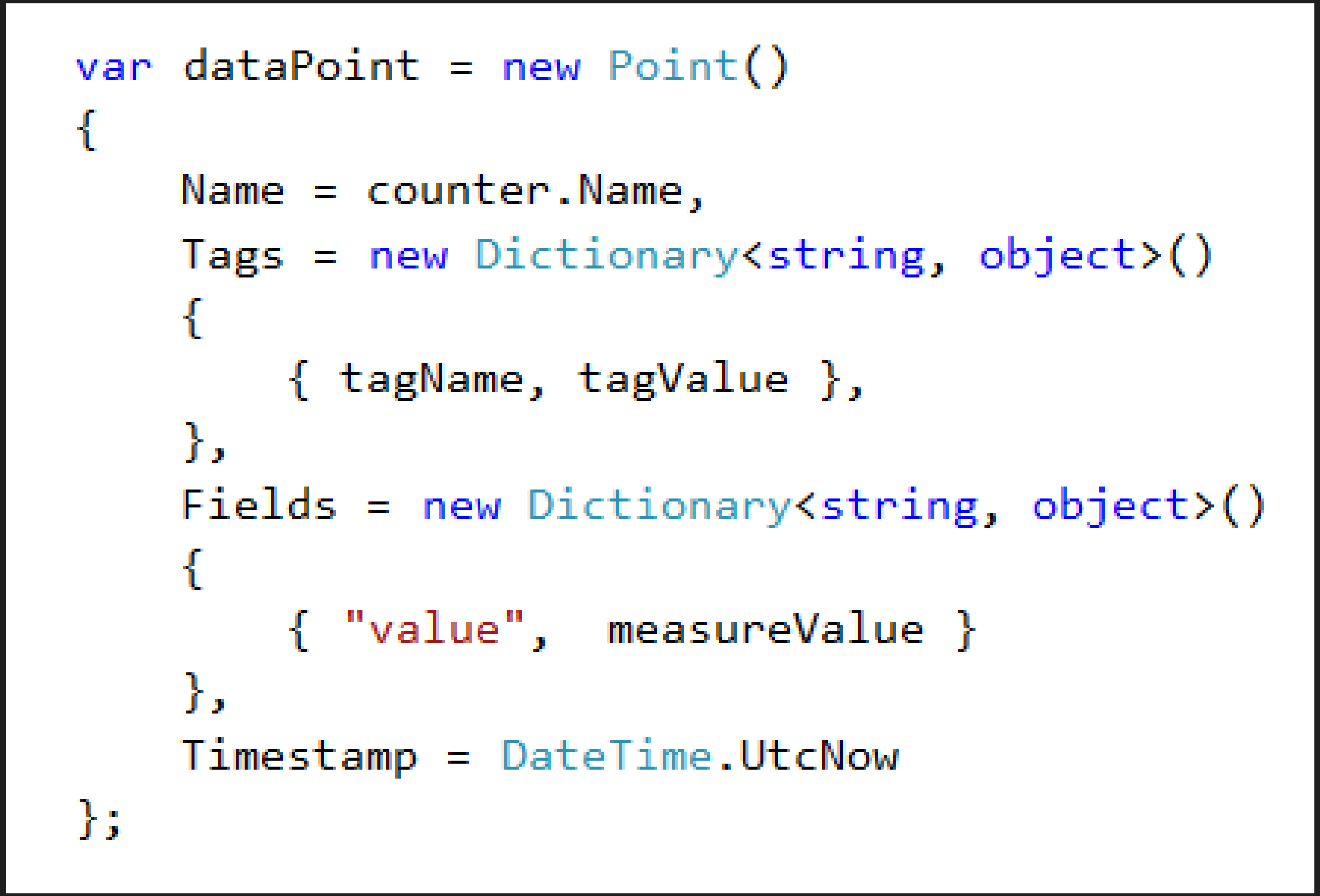
BIGBEARS.IO



Where to measure

BIGBEARS.IO

How to Report



```
var dataPoint = new Point()
{
    Name = counter.Name,
    Tags = new Dictionary<string, object>()
    {
        { tagName, tagValue },
    },
    Fields = new Dictionary<string, object>()
    {
        { "value",  measureValue }
    },
    Timestamp = DateTime.UtcNow
};
```

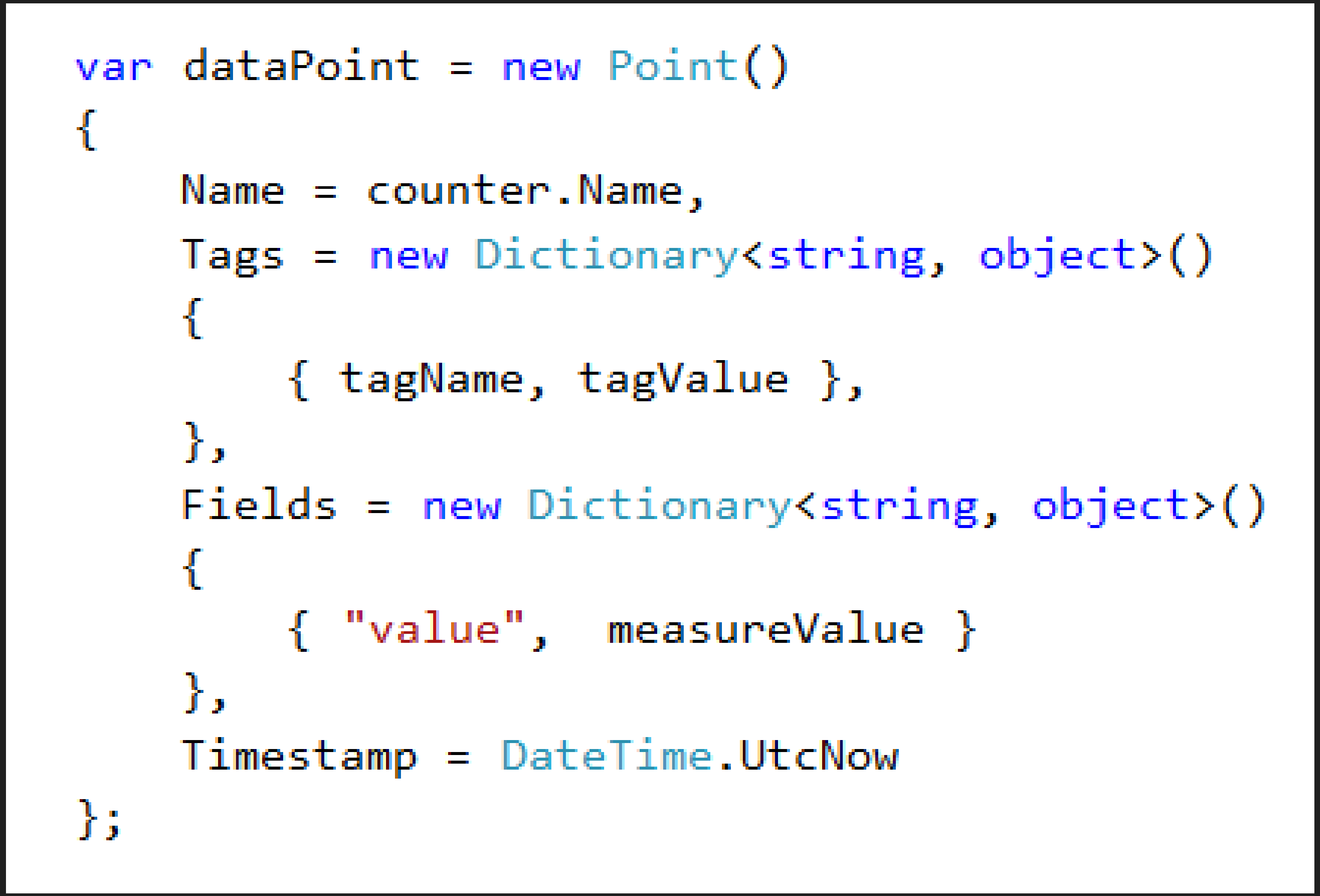
<- Measurement Name

<- Tags and its values

<- Fields and its values

<- Point of time

BIGBEARS.IO



```
var dataPoint = new Point()
{
    Name = counter.Name,
    Tags = new Dictionary<string, object>()
    {
        { tagName, tagValue },
    },
    Fields = new Dictionary<string, object>()
    {
        { "value",  measureValue }
    },
    Timestamp = DateTime.UtcNow
};
```

<- Measurement Name

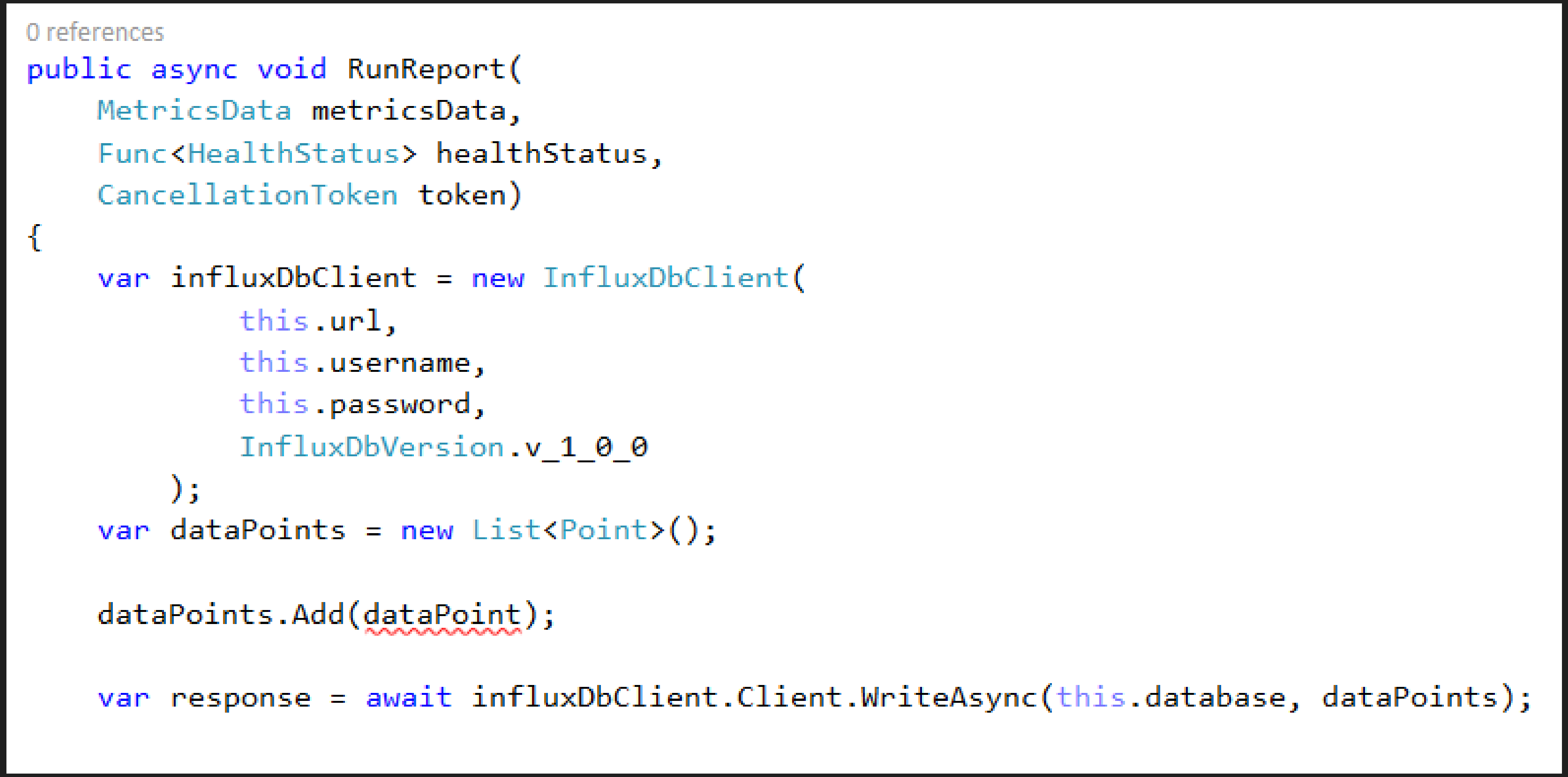
<- Tags and its values

<- Fields and its values

<- Point of time

BIGBEARS.IO

How to Report


```
0 references
public async void RunReport(
    MetricsData metricsData,
    Func<HealthStatus> healthStatus,
    CancellationToken token)
{
    var influxDbClient = new InfluxDbClient(
            this.url,
            this.username,
            this.password,
            InfluxDbVersion.v_1_0_0
        );
    var dataPoints = new List<Point>();

    dataPoints.Add(dataPoint);

    var response = await influxDbClient.Client.WriteAsync(this.database, dataPoints);
```

<- Custom Reporter

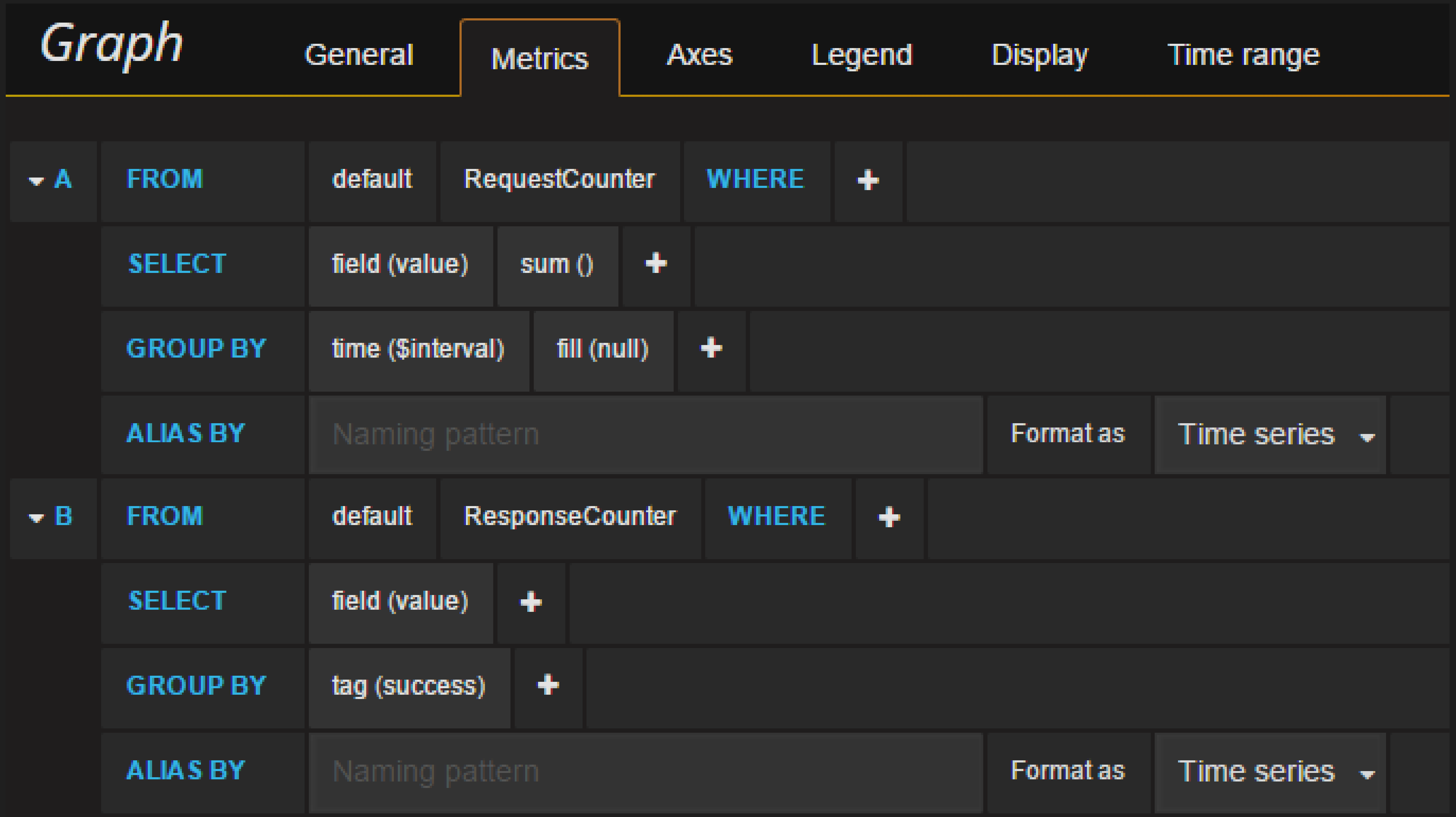
<- Connect to InfluxDb

<- Collect Data points

<- Send to InfluxDb

BIGBEARS.IO

How to Report



Graph — General | Metrics | Axes | Legend | Display | Time range

```
A  FROM      default  RequestCounter  WHERE  +
   SELECT    field (value)  sum ()  +
   GROUP BY  time ($interval)  fill (null)  +
   ALIAS BY  Naming pattern          Format as  Time series
B  FROM      default  ResponseCounter  WHERE  +
   SELECT    field (value)  +
   GROUP BY  tag (success)  +
   ALIAS BY  Naming pattern          Format as  Time series
```

<- Measurement Name

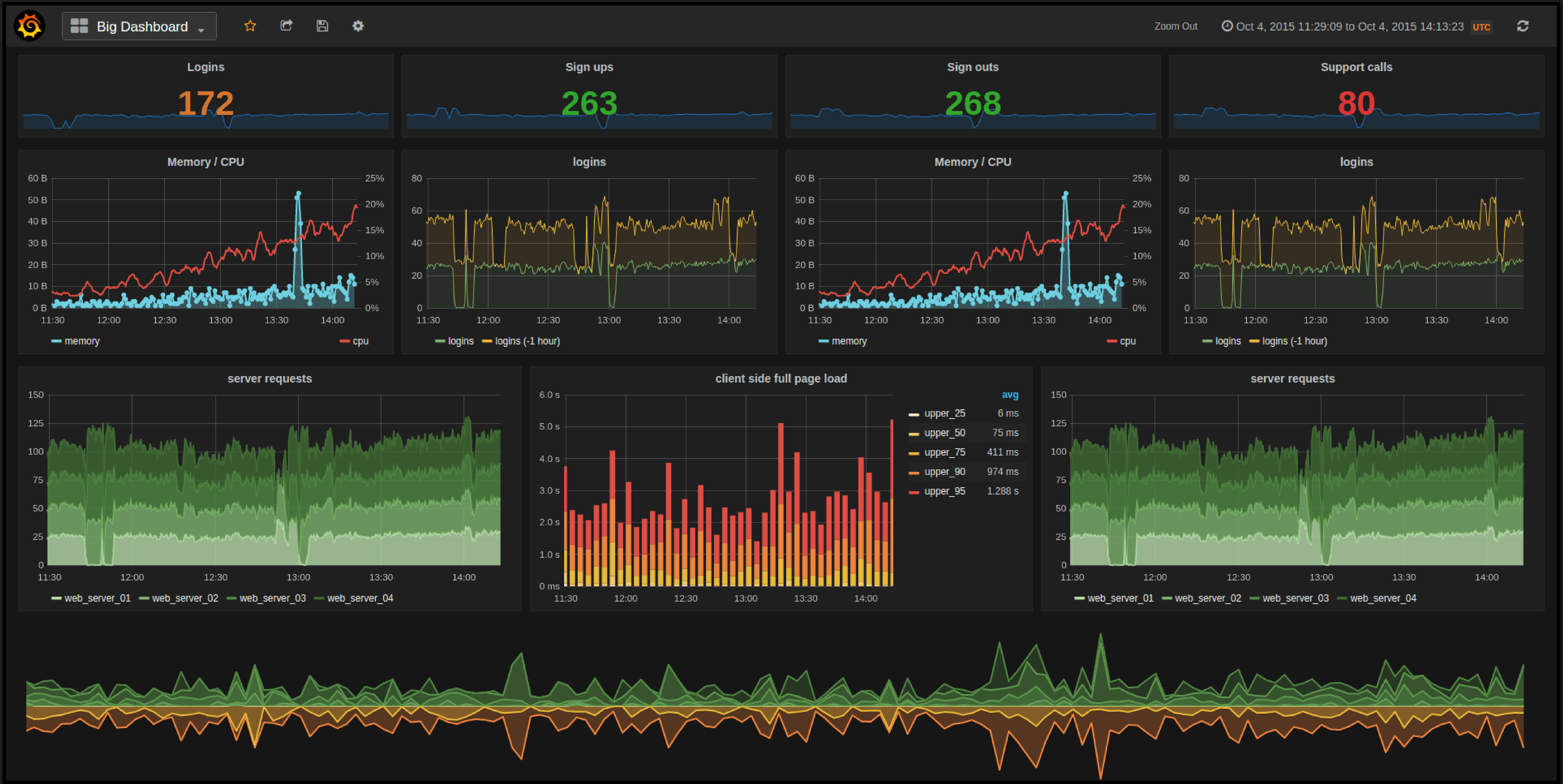
<- Fields and aggregation

<- Tags and its values

<- Alias

BIGBEARS.IO

How to use Metrics



BIGBEARS.IO

How to use Metrics

DEMONSTRATION

https://github.com/bigbearsio/GrafanaDemo

BIGBEARS.IO