# Own your software by using Micrometer and Spring Boot


Ko Turk

# You’re part of a racing team

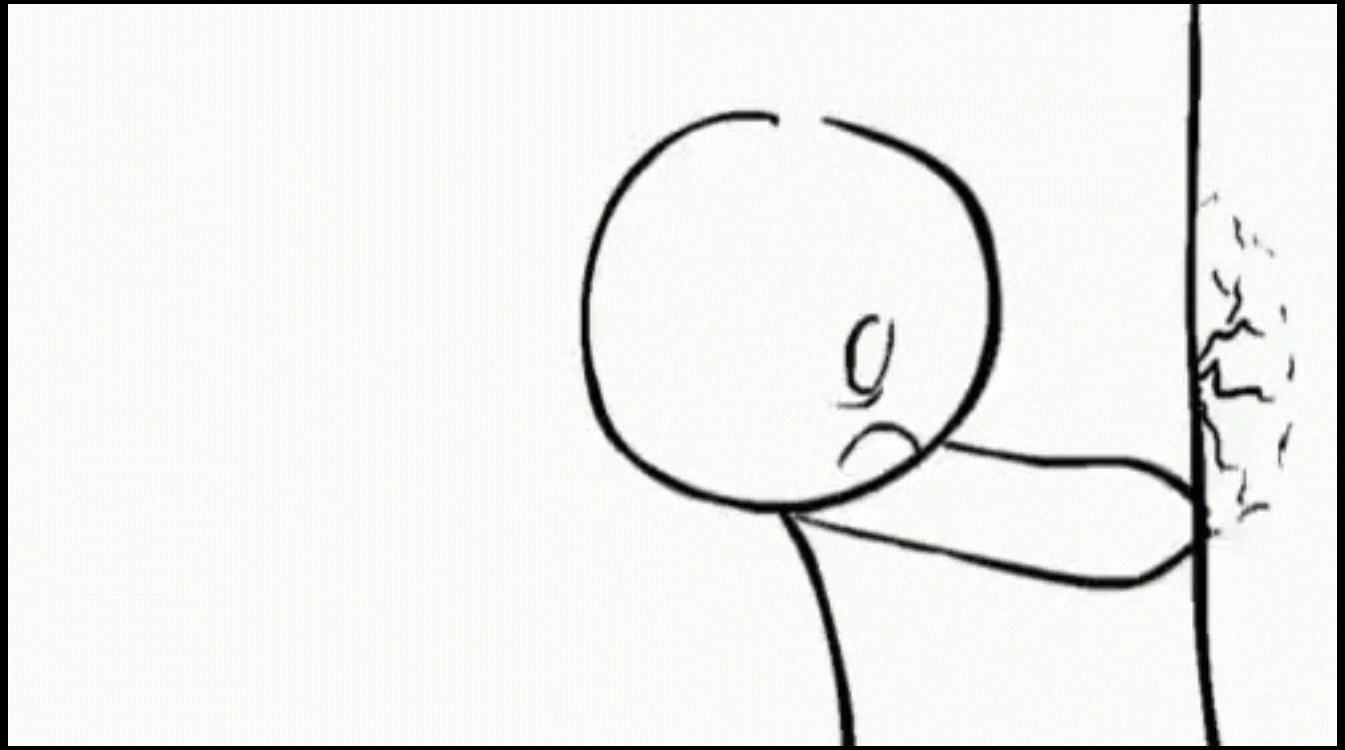

@REDBULLRACING
edBull
Mobil 1
LAT IMAGES

But what if we don’t have this kind of information?

**Sounds silly right?**

But why don’t we (developers) think about proper monitoring?


(still driving but crashing)
Application Y
Backend
https://media.nu.nl/m/bxyxodiacvny_wd1280.jpg/verstappen-en-hamilton-crashen-bij-fel-duel-en-vallen-allebei-uit-op-monza.jpg



If that’s happening…..

```xml
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>log4j-over-slf4j</artifactId>
    <scope>provided</scope>
</dependency>
```

# Of course you can log it but…..

```
} catch (Exception exception) {
    log.error("WTF is going wrong here:", exception);
    return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
}
```

```
log.debug("You never going to read this, logging level is INFO", e);
```

https://www.sumologic.com/blog/logs-metrics-overview/

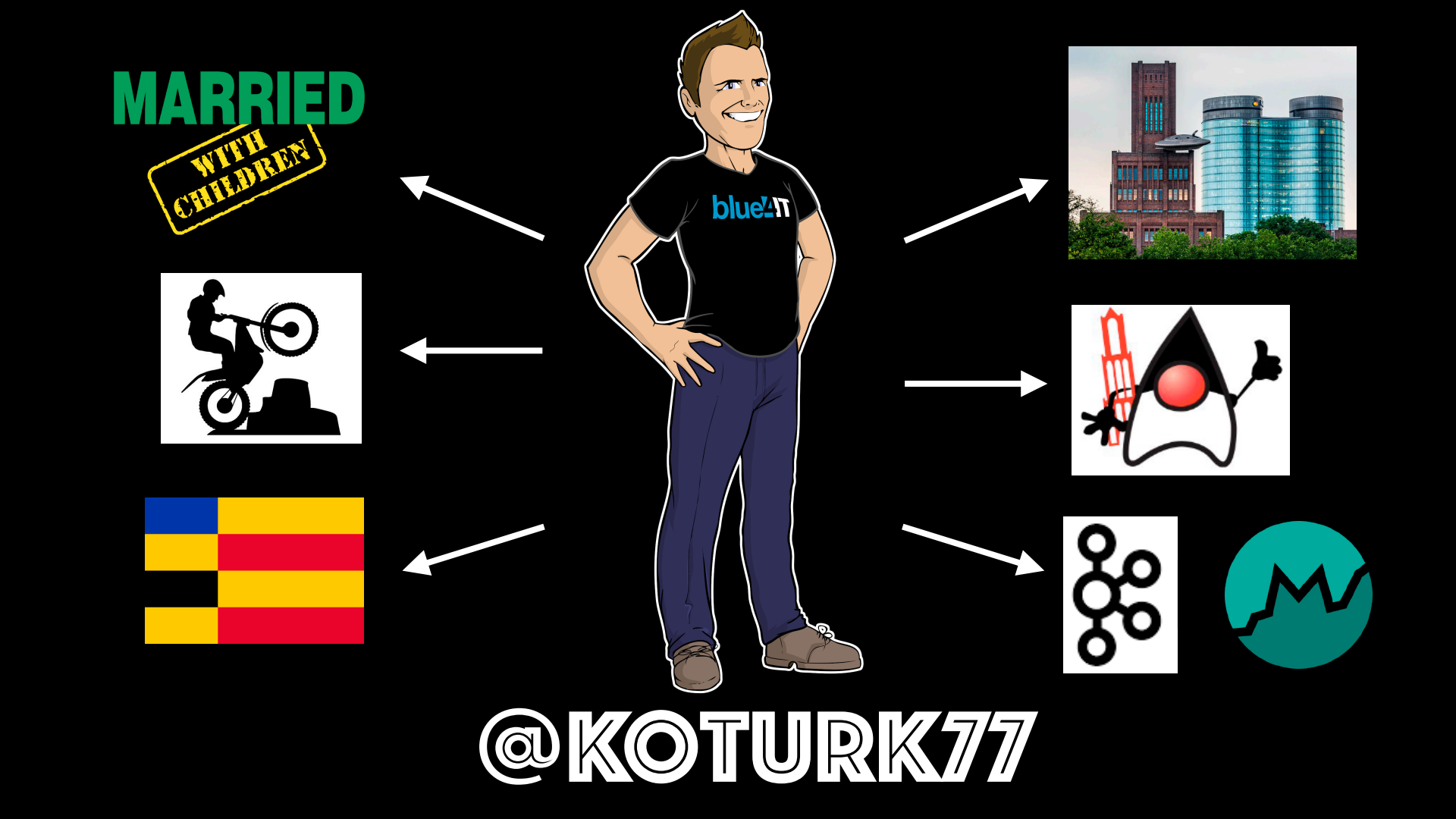
**A log is an event that happened**

**A metric is a measurement of the health of a system.**

That’s where

Micrometer comes in

# Introduction


MARRIED
WITH CHILDREN
blue4IT
@KOTURK77


ORACLE

https://www.menti.com/ysawpti2a7

Mentimeter

Go to **www.menti.com** and use the code **9376 4668**

# Goals

Define the metrics *(Create the teams strategy)*

Create them in Micrometer *(Preparing the car/race)*

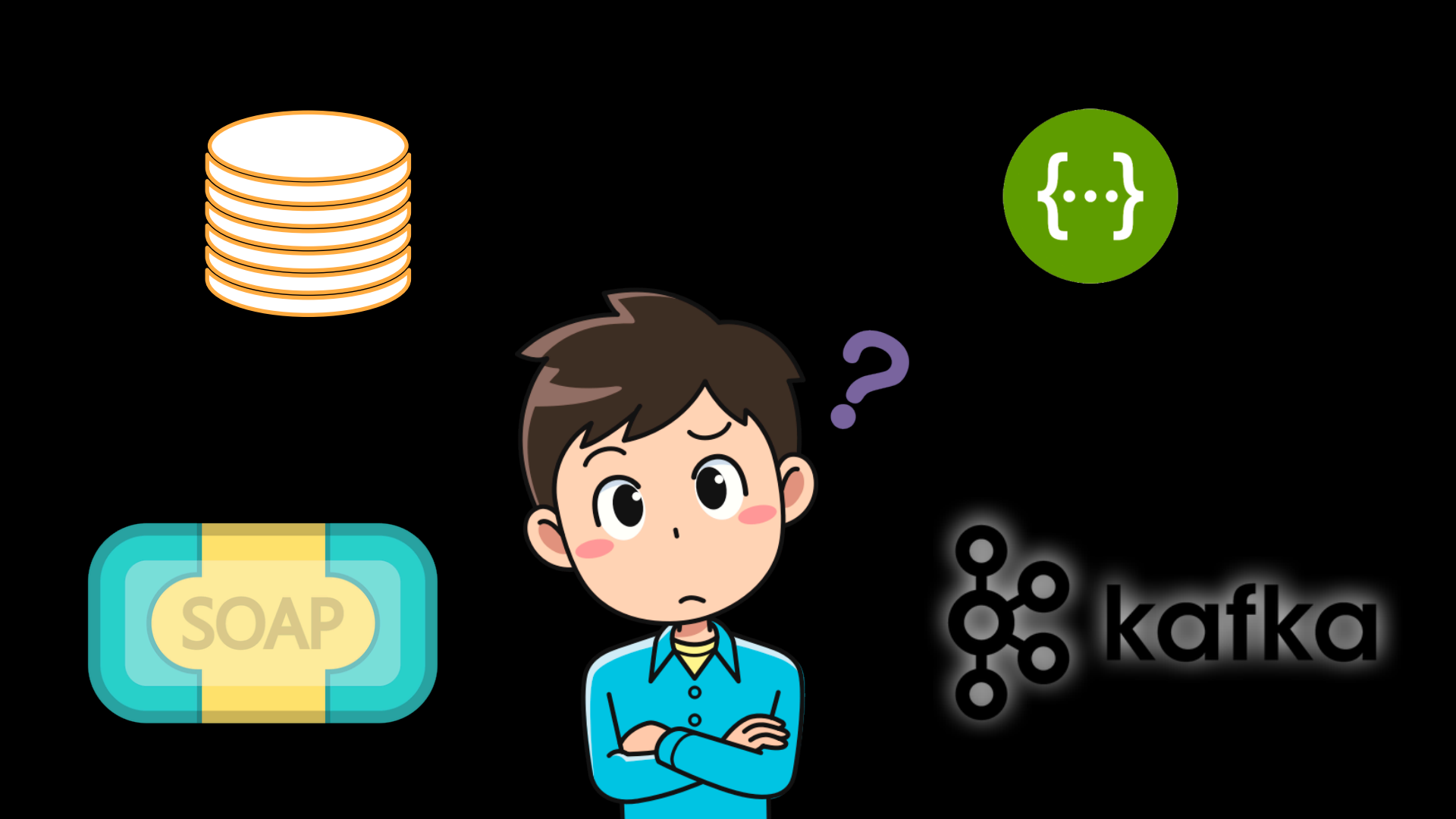
Save it to Prometheus

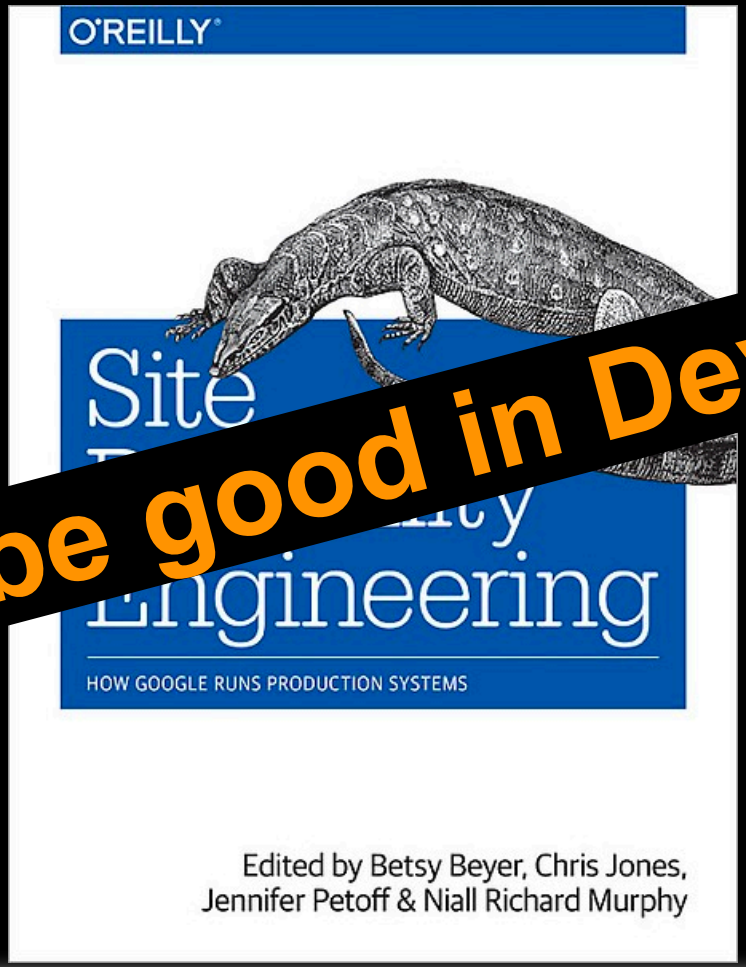
Visualize with Grafana *(The teams Pitwall)*

# Create the teams strategy

https://www.redbull.com/nl-nl/red-bull-racing-zet-de-fabrieksdeuren-open

# What do we want to monitor?


SOAP
kafka




# How to be good in DevOps?


# Site Reliability Engineering

Edited by Betsy Beyer, Chris Jones, Jennifer Petoff and Niall Richard Murphy

Members of the SRE team explain how their engagement with the entire software lifecycle has enabled Google to build, deploy, monitor, and maintain some of the largest software systems in the world.

READ ONLINE FOR FREE

BUY FROM GOOGLE BOOKS

https://landing.google.com/sre/sre-book/toc/index.html

# Racing Factors —> Defining an indicator (SLI)

- pitstop — How long does it take to do a pitstop
  **request latency** — **How long does it take to return a response**

# Tactics —> Defining an objective (SLO)

- pitstop — Shouldn’t take more than 8 seconds
  **request latency** — **Shouldn’t be taking more than 0.1 seconds**

# Strategy —> Defining an agreement (SLA)

- 2 pitstops — less then 8 seconds
  **99,9 % of all requests** — **less than 0.1 seconds**

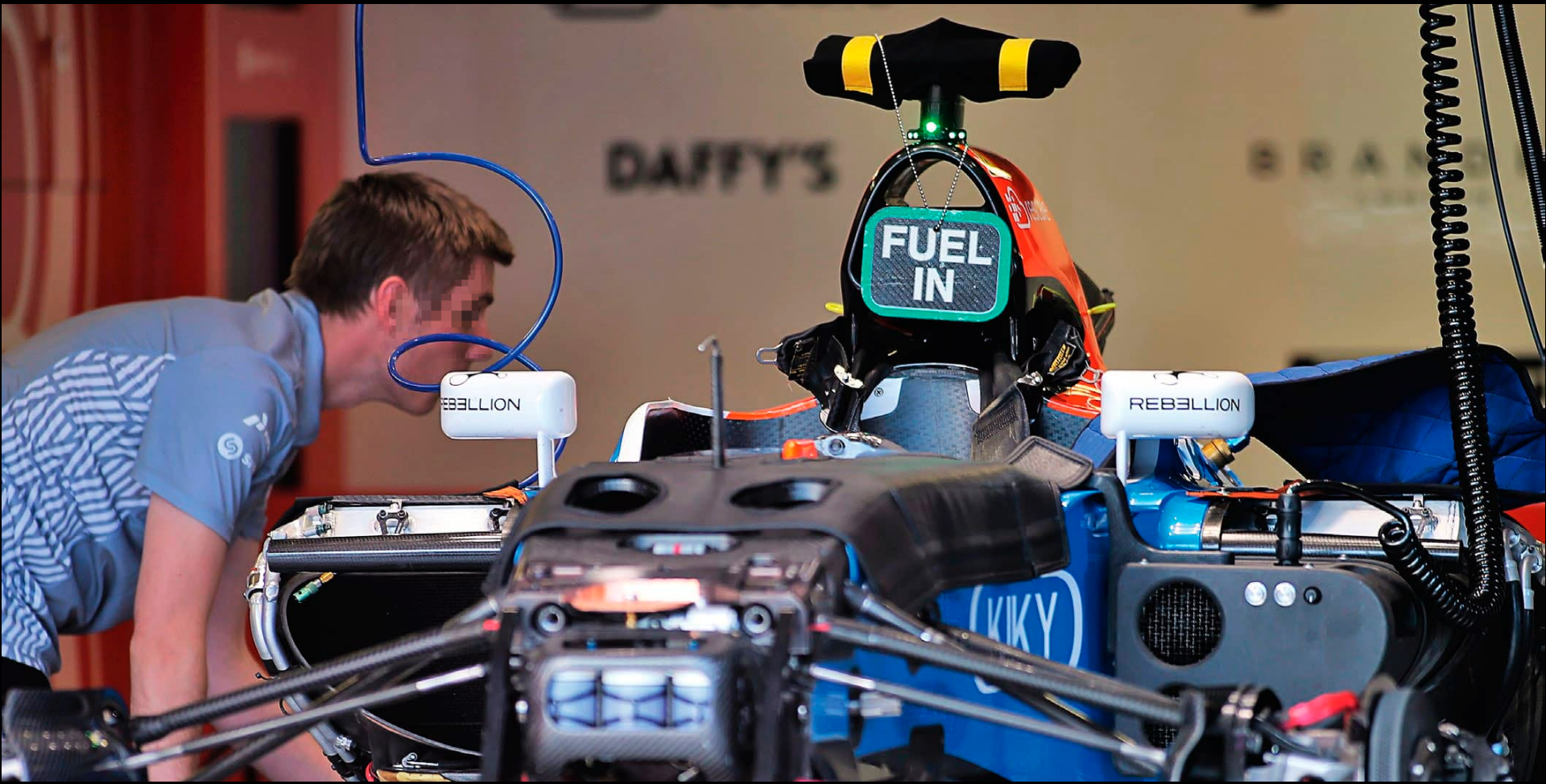
# Goals

~~Define the metrics~~ ~~*(Create the teams strategy)*~~

Create them in Micrometer *(Preparing the car/race)*

Save it to Prometheus

Visualize with Grafana *(The teams Pitwall)*


DAFFY'S
FUEL
IN
REBELLION
REBELLION
KIKY

Micrometer for the win!

**But what is it?**

# SLF4J only for metrics

# Micrometer is included in Spring Boot 2 actuator

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

# Extra dependency for Spring Boot 1.5.x

```
<dependency>
    <groupId>io.micrometer</groupId>
    <artifactId>micrometer-spring-legacy</artifactId>
</dependency>
```

# And also for Micronaut

```xml
<dependency>
    <groupId>io.micronaut.configuration</groupId>
    <artifactId>micronaut-micrometer-core</artifactId>
    <version>${micronaut.version}</version>
</dependency>
```

```xml
<dependency>
    <groupId>io.micronaut</groupId>
    <artifactId>micronaut-management</artifactId>
    <version>${micronaut.management.version}</version>
</dependency>
```

# Pre-configured Bindings

- Instrumentions of:
  - caches
  - class loader
  - garbage collection
  - processor utilisation
  - thread pools
  - and more

# Live coding


https://i.ytimg.com/vi/eVpRNi5VEDo/maxresdefault.jpg

User Interface Theme

# Material Theme UI

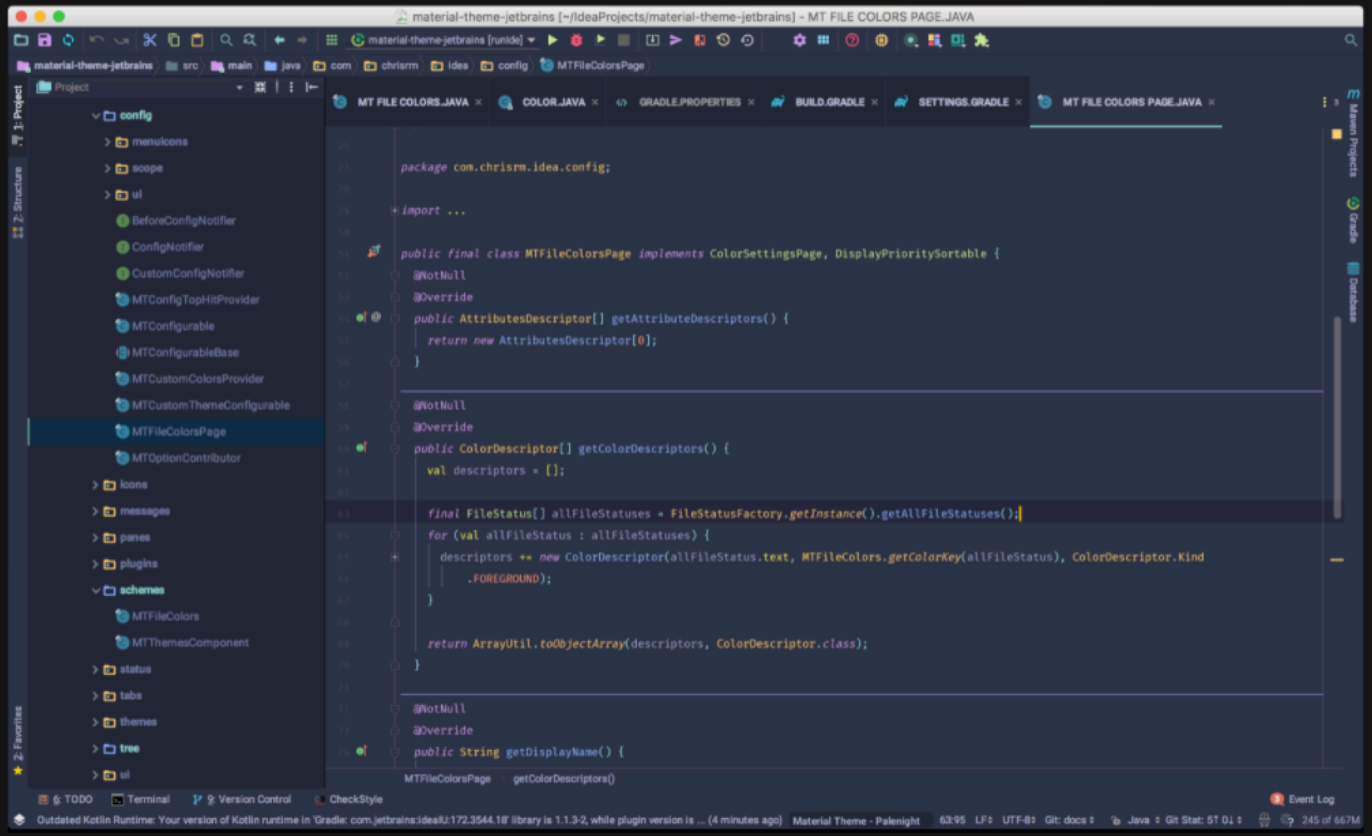
★★★☆☆

Atom Material Themes & Plugins

Install to IntelliJ IDEA 2021.2.1

Some features of the plugin require a paid subscription. Learn more

Compatible with all IntelliJ-based IDEs



# Pre-configured Bindings

- Instrumentions of:
  - caches
  - class loader
  - garbage collection
  - processor utilisation
  - thread pools
  - and more

Go to actuator metrics

# Timer


https://s1.cdn.autoevolution.com/images/news/gallery/red-bull-racing-beats-record-to-set-new-fastest-pit-stop-in-the-history-of-f1_2.jpg

# Long Task Timer


TOTAL
https://i.pinimg.com/originals/fb/5b/98/fb5b98edf68000f44e89d6428a3e2c65.jpg

# Counter


https://www.wallpaperflare.com/static/467/219/735/mercedes-amg-f1-w07-hybrid-formula-1-testing-wallpaper.jpg

# Gauge


https://cdn.24.co.za/files/Cms/General/d/4338/1678467a9d9149b983f38424014fb2c0.jpg

# http.server.requests.count


https://www.formula1.com/content/dam/fom-website/sutton/2018/China/Saturday/dcd1814ap130.jpg

# About security

- Please implement spring security
- Otherwise you will expose your endpoint to others

- And don’t forget to add it in your database config!

```
basic_auth:
  username: "user"
  password: "pass"
```

https://www.baeldung.com/spring-security-basic-authentication
https://egkatzioura.com/2020/05/07/spring-boot-and-micrometer-with-prometheus-part-6-securing-metrics/

# Sensitive data (!)

- Disable endpoints you don’t need

```yaml
management:
  endpoints:
    web:
      exposure:
        include: 'health'
    jmx:
      exposure:
        exclude: '*'
```

# Goals

~~Define the metrics~~ ~~*(Create the teams strategy)*~~

~~Create them in Micrometer~~ ~~*(Preparing the car/race)*~~

Save it to Prometheus

Visualize with Grafana *(The teams Pitwall)*

# Where to store the metrics?

Which database should I use?

# Databases

- Prometheus
- Graphite
- Atlas
- KairosDB

**Persistent or not persistent?**

# If you want to expose to prometheus

```
<dependency>
    <groupId>io.micrometer</groupId>
    <artifactId>micrometer-registry-prometheus</artifactId>
</dependency>
```

# Or Graphite

```
<dependency>
    <groupId>io.micrometer</groupId>
    <artifactId>micrometer-registry-graphite</artifactId>
</dependency>
```

```
# TYPE jvm_buffer_memory_used_bytes gauge
jvm_buffer_memory_used_bytes{id="direct",} 24576.0
jvm_buffer_memory_used_bytes{id="mapped",} 0.0
# HELP process_start_time_seconds Start time of the process since unix epoch.
# TYPE process_start_time_seconds gauge
process_start_time_seconds 1.581090649742E9
# HELP jvm_buffer_total_capacity_bytes An estimate of the total capacity of the buffers in this pool
# TYPE jvm_buffer_total_capacity_bytes gauge
jvm_buffer_total_capacity_bytes{id="direct",} 24576.0
jvm_buffer_total_capacity_bytes{id="mapped",} 0.0
# HELP gauge
# TYPE gauge gauge
gauge 7.0
# HELP logback_events_total Number of error level events that made it to the logs
# TYPE logback_events_total counter
logback_events_total{level="warn",} 0.0
logback_events_total{level="debug",} 0.0
logback_events_total{level="error",} 0.0
logback_events_total{level="trace",} 0.0
logback_events_total{level="info",} 8.0
# HELP counterBuilder_total
# TYPE counterBuilder_total counter
counterBuilder_total{continent="USA",} 0.0
# HELP jvm_memory_committed_bytes The amount of memory in bytes that is committed for the Java virtual machine to use
# TYPE jvm_memory_committed_bytes gauge
jvm_memory_committed_bytes{area="heap",id="PS Survivor Space",} 1.3631488E7
jvm_memory_committed_bytes{area="heap",id="PS Old Gen",} 1.76160768E8
jvm_memory_committed_bytes{area="heap",id="PS Eden Space",} 1.42606336E8
jvm_memory_committed_bytes{area="nonheap",id="Metaspace",} 3.9452672E7
jvm_memory_committed_bytes{area="nonheap",id="Code Cache",} 9568256.0
jvm_memory_committed_bytes{area="nonheap",id="Compressed Class Space",} 5636096.0
# 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 system_cpu_count The number of processors available to the Java virtual machine
# TYPE system_cpu_count gauge
system_cpu_count 16.0
# HELP gaugeSimple
# TYPE gaugeSimple gauge
gaugeSimple 7.0
# HELP jvm_threads_peak_threads The peak live thread count since the Java virtual machine started or peak was reset
# TYPE jvm_threads_peak_threads gauge
jvm_threads_peak_threads 29.0
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/metrics",} 1.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/metrics",} 0.017340408
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/metrics",} 0.017340408
# HELP process_files_max_files The maximum file descriptor count
# TYPE process_files_max_files gauge
process_files_max_files 10240.0
# HELP gaugeBuilder
# TYPE gaugeBuilder gauge
gaugeBuilder 7.0
```

# Goals

~~Define the metrics~~

~~*(Create the teams strategy)*~~

~~Create them in Micrometer~~

~~*(Preparing the car/race)*~~

~~Save it to Prometheus~~

Visualize with Grafana

*(The teams Pitwall)*

# PURE PITWALL

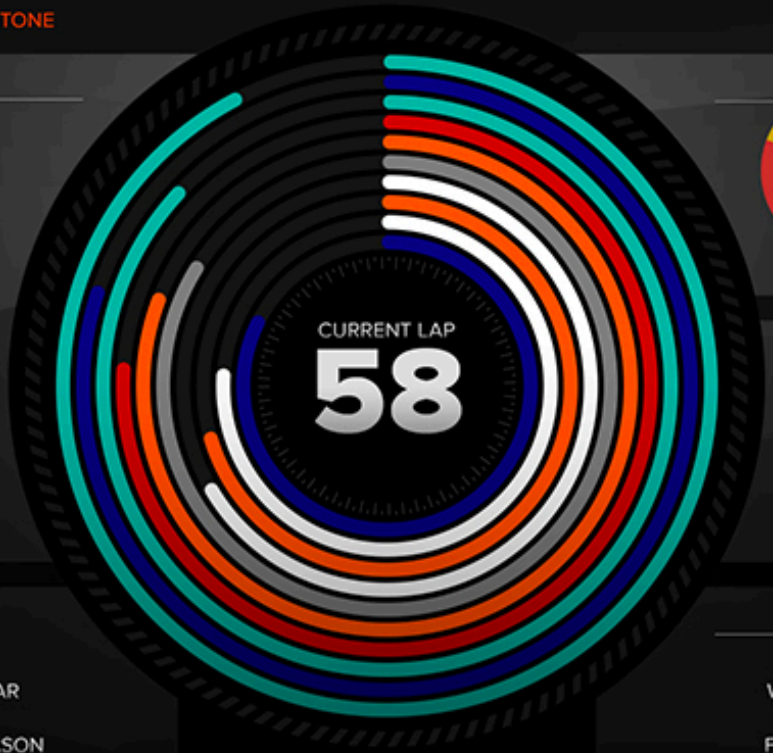
OFFICIAL TEAM PARTNER

2016 FORMULA 1 BRITISH GRAND PRIX // SILVERSTONE

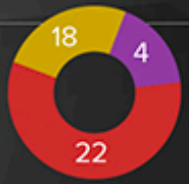
TRACK TEMP: 29°C AIR TEMP: 20°C

## PIT STOP TIMES

| | |
|---|---|
| ROS | FASTEST |
| VET | +0.51 |
| BUT | +0.74 |
| BOT | +0.78 |
| RIC | +0.90 |
| HUL | +1.12 |
| HAM | +1.19 |
| MAS | +1.34 |
| ALO | +1.95 |
| RAI | +2.40 |




## TYRE COMPOUNDS




SETS USED

- SUPER SOFT
- ULTRA SOFT
- SOFT

ROS 1:40.418 **FASTEST LAPS:** 10/67

**TRACK RECORD:** 1:39.768

## SAFETY CAR LAPS

**07**

4 LAST YEAR

28 THIS SEASON

## TOTAL OVERTAKES

WITH DRS 28

FOR LEAD 2

**35**

https://blog.purestorage.com/wp-content/uploads/2016/07/Pure-MBF1-Pitwall-v6.jpg

# How to visualise the application metrics?

Which monitoring system to use?

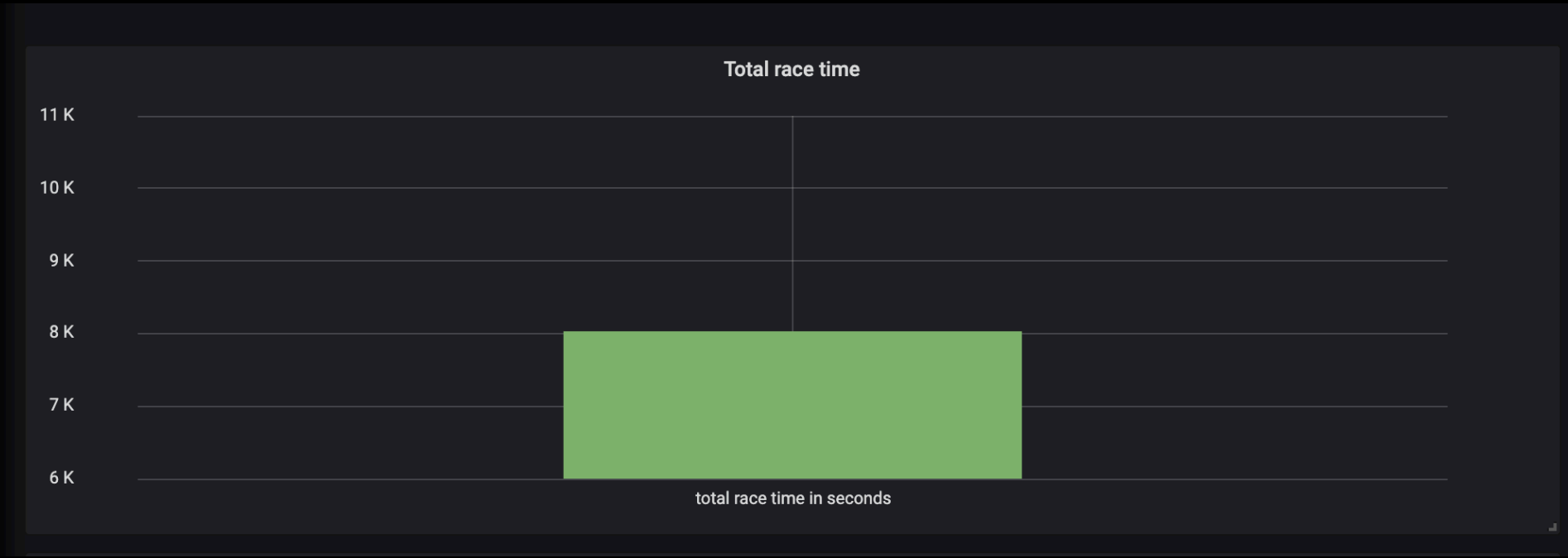
## Systems

- Grafana
- Datadog
- Dynatrace
- Instant
- WaveFront

- and many more

# Let’s deep dive into Grafana

# Long Task Timer (duration)



# Timer (count)

# Timer (max)

# Gauge

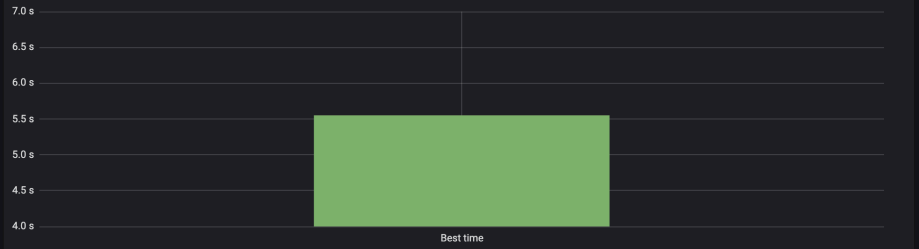
# Bindings

Race-Demo

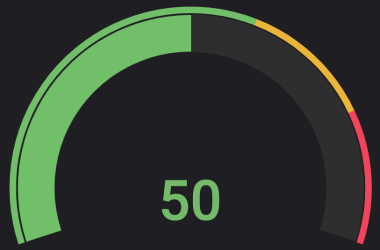
Last 6 hours

## Pitstops


## Pitstop




## Tire wear in %




## Total race time

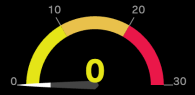

## Cars in race (thread count)

20

[redacted] sum calls 12h

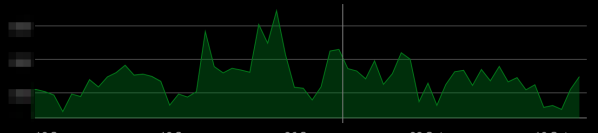

[redacted] error/min 10s

error_count/((error_count+success_count)/100)

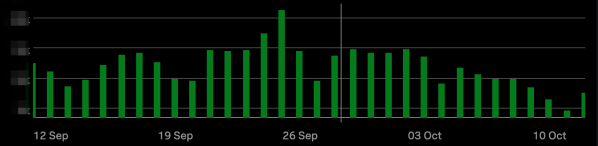



Tue 12 Oct 2021 00:00:10

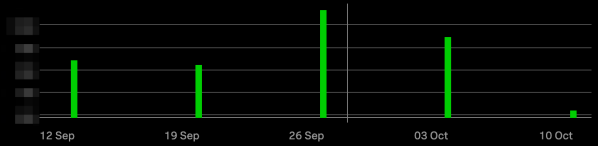
[redacted] sum calls 12h




Times [redacted] Page (daily) 12h




Times Loaded [redacted] Page (weekly) 12h




Times Loaded [redacted] Page (monthly) 12h


[redacted] error responses sum per day 2m

| | |
|---|---|
| 11 | [redacted]xception \| PUT \| SERVER_ERROR \| 500 |
| 9 | [redacted] Exception \| PUT \| CLIENT_ERROR \| 400 |
| 22 | [redacted]Exception \| PUT \| SERVER_ERROR \| 500 |

Tue 12 Oct 2021 00:00:00

[redacted] responses count per day 2m

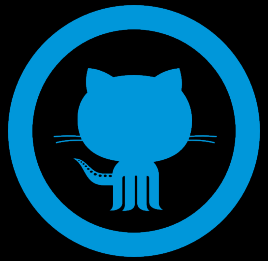
| | |
|---|---|
| 55 | [redacted] Exception \| PUT \| SERVER_ERROR \| 500 |
| 0 | [redacted]Exception \| GET \| CLIENT_ERROR \| 400 |
| 9 | [redacted]Exception \| GET \| CLIENT_ERROR \| 403 |
| 20 | [redacted]Exception \| PUT \| SERVER_ERROR \| 500 |

[redacted] error responde calls per day 2m

| | |
|---|---|
| 0 | [redacted]Exception \| POST \| [redacted] |


FIA
Red Bull
ESSO
44

# Thank you for listening



All examples you can find at:
https://github.com/KoTurk/

@KoTurk77