

Как количество тестов производительности переходит в качество

Спустя годы попыток

**Спустя годы попыток
отчеты по нагрузке**

Спустя годы попыток
отчеты по нагрузке
интересны читателям



Спустя годы попыток

отчеты по нагрузке

интересны читателям

Автотесты

Автотесты

производительности.

Автотесты

производительности,

которых много.

Автотесты

производительности,

которых много,

и это хорошо



Смирнов Вячеслав

Ускоряю miro.com

Развиваю [@qa_load](https://www.instagram.com/@qa_load)

План

1. Чем сложна нагрузка?
2. Как понять результат теста за секунду?
3. Как понять сотню результатов тестов за минуту?
4. Что дает Test Management System (**TMS**)?
5. Что дает Test Framework (**JUnit**) писателю?
6. Почему **больше** тестов – **лучше**?

1

Чем сложна нагрузка?

Просто ли

написать

нагрузочные тесты?

А все запросы
учтены?

Что по
ресурсам?

Предупредили
безопасников?

А это
пиковый час?

Места
хватит?

Персональных
данных нет?

Профиль
совпадает?

Инструмент
тянет?

Обработка
ошибок есть?

Мониторинг
настроен?

Лимиты
настроены?

А это 100к
пользователей?

А все запросы
учтены?

А это пиковый
час?

Профиль
совпадает?

Мониторинг
настроен?

Что по
ресурсам?

Места
хватит?

Инструмент
тянет?

Лимиты
настроены?

Предупредили
безопасников?

Персональных
данных нет?

Обработка
ошибок есть?

А это 100к
пользователей?

А все запросы
учтены?

А это
пиковый час?

Что по
ресурсам?

Предупредили
безопасников?

Профиль
совпадает?

Места
хватит?

Персональных
данных нет?

Инструмент
тянет?

Мониторинг
настроен?

Лимиты
настроены?

Обработка
ошибок есть?

А это 100к
пользователей?

А все запросы
учтены?

А это
пиковый час?

Профиль
совпадает?

Мониторинг
настроен?

Что по
ресурсам?

Места
хватит?

Инструмент
тянет?

Лимиты
настроены?

Предупредили
безопасников?

Персональных
данных нет?

Обработка
ошибок есть?

А это 100к
пользователей?

А все запросы
учтены?

А это
пиковый час?

Профиль
совпадает?

Мониторинг
настроен?

Что по
ресурсам?

Места
хватит?

Инструмент
тянет?

Лимиты
настроены?

Предупредили
безопасников?

Персональных
данных нет?

Обработка
ошибок есть?

А это 100к
пользователей?

Непростые

тесты

Просто ли

понять отчет

по нагрузке?

100
графиков

100
ГБайт
логов

10
ГБайт
трейсов

5 попыток
запуска
теста

5 попыток
запуска
теста

10000
ошибок

1000
отличий

10
гипотез

Что будет с продуктивом под нагрузкой?

100
графиков

100
Гбайт

10
Гбайт
графиков

5 по-
запросов

5 по-
запросов

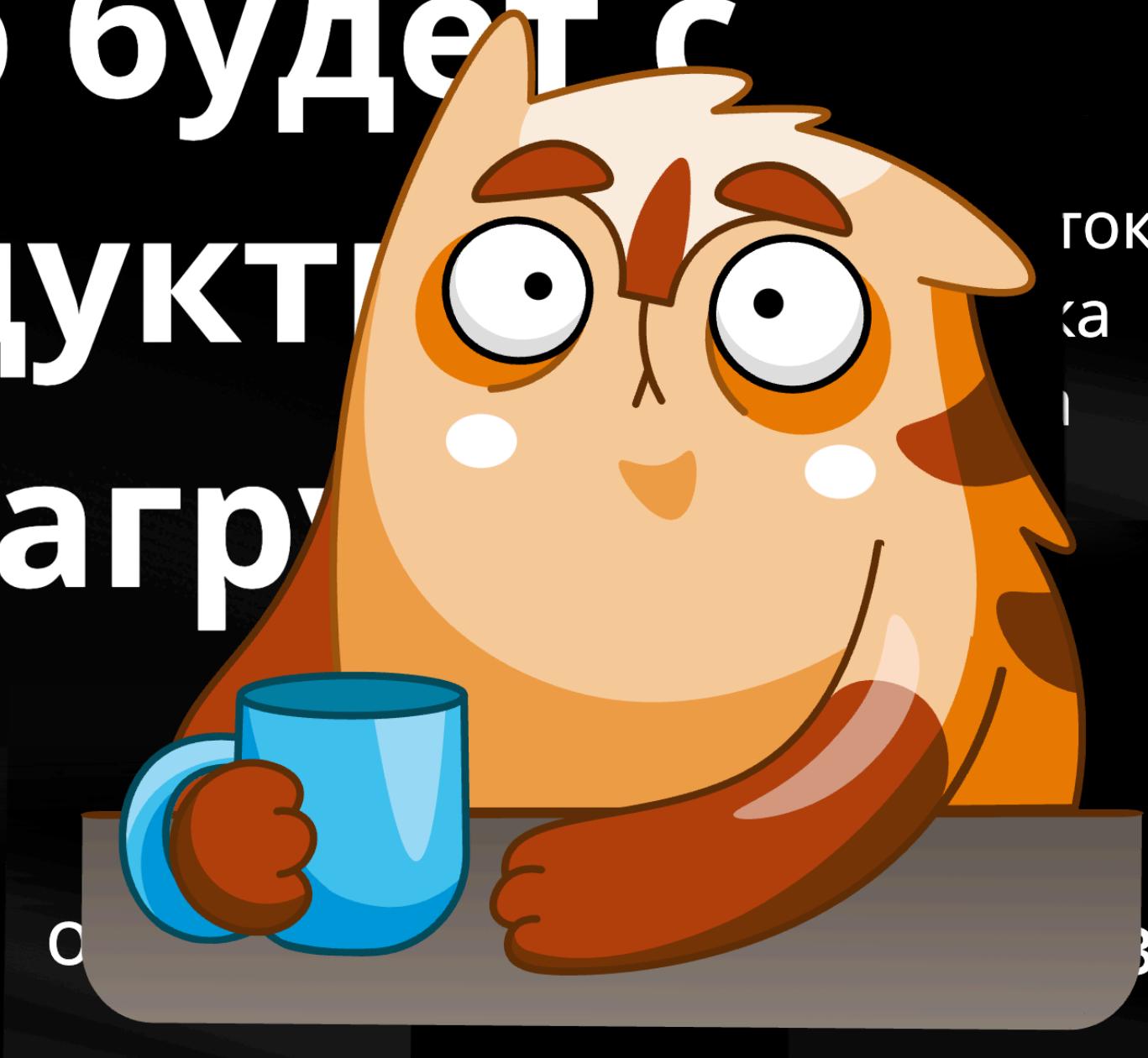
10000
ошибок

1000
отличий

10
гипотез

100
ГБайт
10
ГБайт
100
графиков
5 попыток
запуска
10000
ошибок

Что будет с
продуктом
под нагрузкой?



Непростые

тесты и

результаты

Непростые

тесты и

результаты

людям

трудно понимать

результаты нагрузки

2

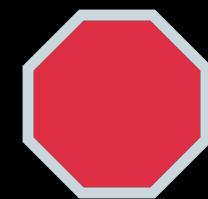
Как понять

результат теста

за секунду?

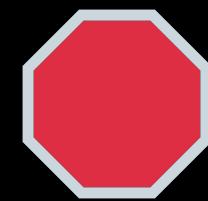
Результат **одного** теста:

 OK

 Fail

Результат **одного** теста:

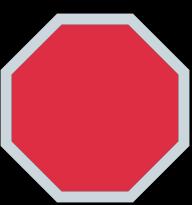
 **OK**

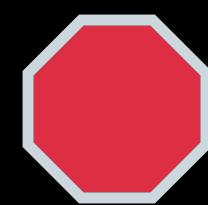
 **Fail**

 **не получилось проверить**

Результат набора тестов:

 15

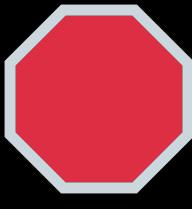
 10  5

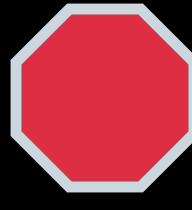
 15

 15

Результат набора тестов:

 15 – все хорошо

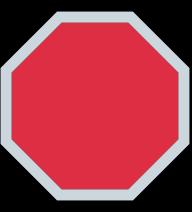
 10  5

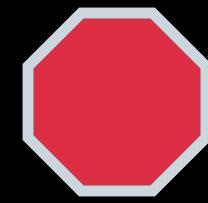
 15

 15

Результат набора тестов:

 15

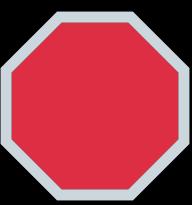
 10  5 – **кое-что сломалось**

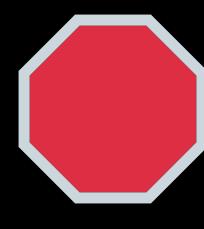
 15

 15

Результат набора тестов:

 15

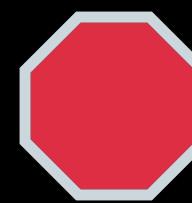
 10  5

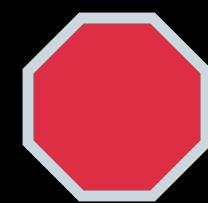
 15 – **все плохо**

 15

Результат набора тестов:

 15

 10  5

 15

 15 – не удалось проверить

Что будет с продуктивом под нагрузкой?

100
графиков

100
Гбайт

10
Гбайт
графиков

5 по-
запросов

5 по-
запросов

10000
ошибок

1000
отличий

10
гипотез

OK

Fail

OK

OK

OK

Fail

OK

OK

OK

OK

Fail

OK

Мои изменения
ничего не
сломали?

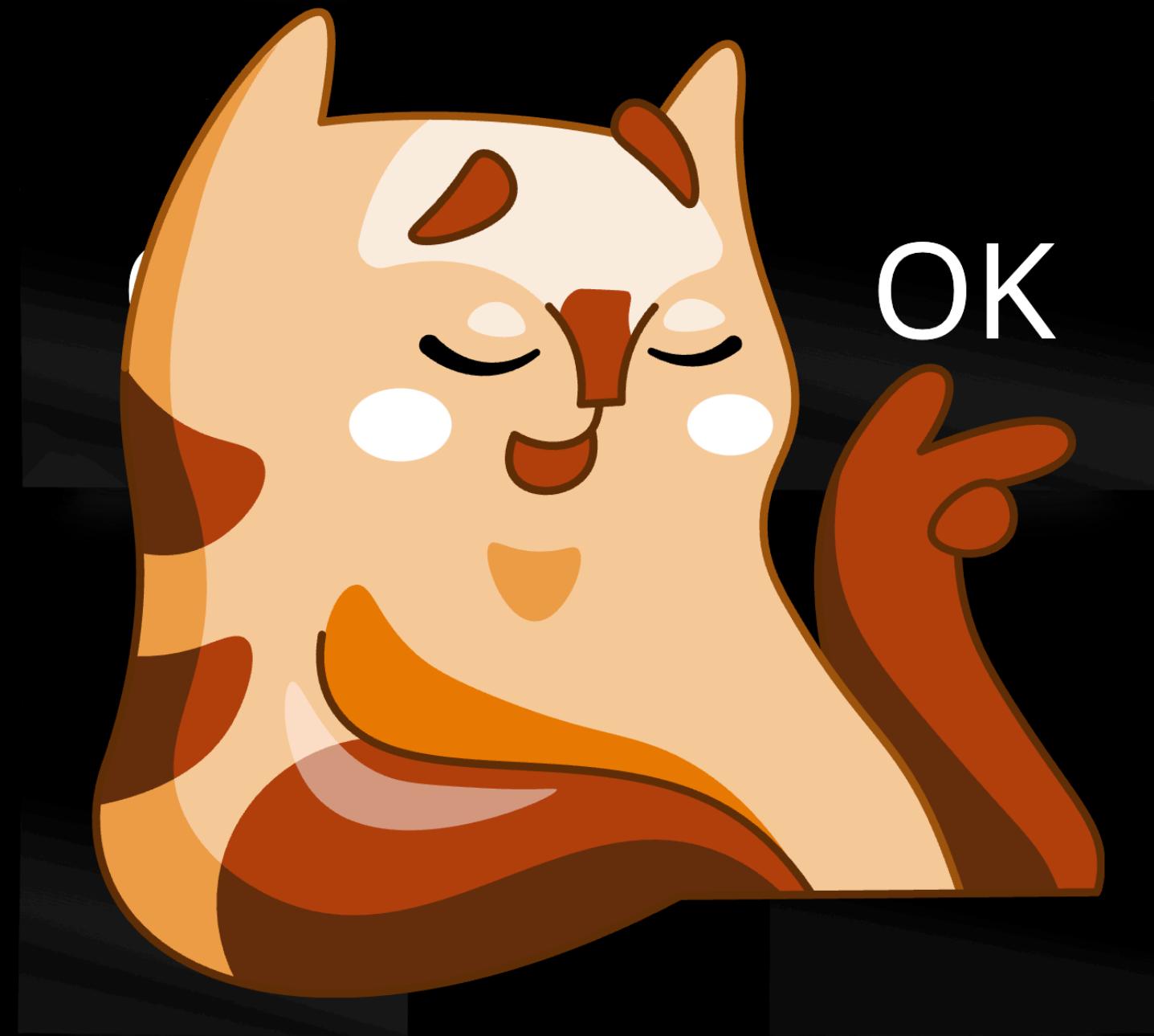
OK

OK

OK

OK

Fail

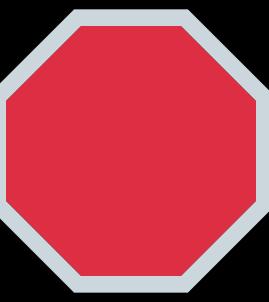


OK

Fail

OK

В результате важны

статус:  

и описание: сервис, версия, ...

3 Как

сделать анализ

сотни результатов?

Что с сервисами?

Service	Version 1.2.3	Version 1.2.4
API Service	 250	 242  8

Что с сервисами?

Service	Version 1.2.3	Version 1.2.4
API Service	 250	 242  8

В **API Service** что-то сломалось в версии **1.2.4**

Кликаем по Service API Service

Service	Version 1.2.3	Version 1.2.4
API Service	 250	 242  8

и переходим в детали

Что с API?

Endpoint	Version 1.2.3	Version 1.2.4
GET /project	 50	 42  8
GET /users	 100	 100
GET /account	 100	 100

Сбой в GET /project

Endpoint	Version 1.2.3	Version 1.2.4
GET /project	✓ 50	✓ 42 ✗ 8
GET /users	✓ 100	✓ 100
GET /account	✓ 100	✓ 100

Кликаем по `GET /project`

Endpoint	Version 1.2.3	Version 1.2.4
<code>GET /project</code>	✓ 50	✓ 42 8
<code>GET /users</code>	✓ 100	✓ 100
<code>GET /account</code>	✓ 100	✓ 100

ЧТО НЕ ТАК С GET /project?

Endpoint	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=name	10	2 8
GET /project	?fields=items	20	20
GET /project	?fields=permissions	20	20

Поле **name** замедлилось

Endpoint	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields= name	✓ 10	✓ 2 8
GET /project	?fields=items	✓ 20	✓ 20
GET /project	?fields=permissions	✓ 20	✓ 20

Кликаем по payload name ...

Endpoint	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=name	10	2 8
GET /project	?fields=items	20	20
GET /project	?fields=permissions	20	20

Проанализировали результаты

- Service: **API Service**
- Endpoint: **GET /project**
- Payload: **?fields=name**
- Version: **1.2.4**

Проанализировали результаты

- Service: **API Service**
- Endpoint: **GET /project**
- Payload: **?fields=name**
- Version: **1.2.4**

за несколько кликов

Таблицы

преобразуют статусы в

аналитический отчет

4

Что дает

Test Management
System (**TMS**)?

Это **подход**

для разных ТМС:

Allure, Qase, TestRail,
TestIT, самодельных ...

Это подход

для разных ТМС:

Allure, Qase, TestRail,

TestIT, самодельных ...

OK

Fail



OK

Fail

OK

OK

Fail

OK

OK

OK

Fail

OK



Можно сделать сводный отчет

Endpoint	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=name	10	2 8
GET /project	?fields=items	20	20
GET /project	?fields=permissions	20	20

Где есть **группировка** по

- Service
- Endpoint
- Payload
- Version
- ...

Introduction

Install & Upgrade

Guides

How it works

Integrations

Frameworks

Behat

Behave

Codeception

CodeceptJS

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

Getting started

Configuration

Reference

Mocha

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

allurereport.org/docs/junit5/#organize-tests

To specify a test's location in the [behavior-based hierarchy](#):

Annotations API Runtime API

```
import io.qameta.allure.Epic;
import io.qameta.allure.Feature;
import io.qameta.allure.Story;
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {

    @Test
    @Epic("Web interface")
    @Feature("Essential features")
    @Story("Authentication")
    void testAuthentication() {
        // ...
    }
}
```



To specify a test's location in the [suite-based hierarchy](#):

java

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {
```

```
    @Test
    void testAuthentication() {
```

```
        Allure.label("parentSuite", "Tests for web interface");
    }
}
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

Install & Upgrade

Features

Guides

Frameworks

Behat

Behave

Codeception

CodeceptJS

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

Getting started

Configuration

Reference

Mocha

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

allurereport.org/docs/junit5/#organize-tests

To specify a test's location in the [behavior-based hierarchy](#):

Behavior: Epic / Feature / Story

```
Annotations API Runtime API  
import io.qameta.allure.Epic;  
import io.qameta.allure.Feature;  
import io.qameta.allure.Story;  
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {  
  
    @Test  
    @Epic("Web interface")  
    @Feature("Essential features")  
    @Story("Authentication")  
    void testAuthentication() {  
        // ...  
    }  
}
```

To specify a test's location in the [suite-based hierarchy](#):

```
java  
import io.qameta.allure.Allure;  
import org.junit.jupiter.api.Test;  
  
class TestMyWebsite {  
  
    @Test  
    void testAuthentication() {  
        Allure.label("parentSuite", "Tests for web interface");  
    }  
}
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

Install & Upgrade

Features

>

Guides

How to work

>

Integrations

>

Frameworks

>

allurereport.org/docs/junit5/#organize-tests

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

Behavior: Epic / Feature / Story

To specify a test's location in the [behavior-based hierarchy](#):

```
Annotations API Runtime API  
Import io.qameta.allure.Epic;  
import io.qameta.allure.Feature;  
import io.qameta.allure.Story;  
import org.junit.jupiter.api.Test;
```

Annotations API

```
@Test  
@Epic("Web interface")  
@Feature("Essential features")  
@Story("Authentication")  
void testAuthentication() {  
    // ...  
}
```

To specify a test's location in the [suite-based hierarchy](#):

```
java  
import io.qameta.allure.Allure;  
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {
```

```
    @Test  
    void testAuthentication() {
```

```
        Allure.label("parentSuite", "Tests for web interface");
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

Install & Upgrade

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

allurereport.org/docs/junit5/#organize-tests

To specify a test's location in the [behavior-based hierarchy](#):

Behavior: Epic / Feature / Story

Annotations API

**@Epic("Web interface")
@Feature("Essential features")
@Story("Authentication")**

To specify a test's location in the [suite-based hierarchy](#):

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {
```

```
    @Test
    void testAuthentication() {
```

```
        Allure.label("parentSuite", "Tests for web interface");
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

[Install & Upgrade](#) >

[Getting started](#) >

[Features](#) >

[Guides](#) >

[How it works](#) >

[Integrations](#) >

[Frameworks](#) >

Behat >

Behave >

Codeception >

CodeceptJS >

Cucumber.js >

Cucumber-JVM >

Cucumber.rb >

Cypress >

Jasmine >

JBehave >

Jest >

JUnit 4 >

JUnit 5 >

[Getting started](#)

Configuration

Reference

Mocha >

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the relevant fields to tests either by adding annotations or "dynamically" (same as for the [metadata fields](#)).

To specify a test's location in the [behavior-based hierarchy](#):

[Annotations API](#)

[Runtime API](#)

```
import io.qameta.allure.Epic;
import io.qameta.allure.Feature;
import io.qameta.allure.Story;
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {

    @Test
    @Epic("Web interface")
    @Feature("Essential features")
    @Story("Authentication")
    void testAuthentication() {
        // ...
    }
}
```



To specify a test's location in the [suite-based hierarchy](#):

java

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {
```

```
    @Test
    void testAuthentication() {
```

```
        Allure.label("parentSuite", "Tests for web interface");
    }
}
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

On this page

[Setting up](#)

[Add Allure dependencies](#)

[Results ...](#)

allurereport.org/docs/junit5/#organize-tests

Guides

How it works

Integrations

Frameworks

Behat

Behave

Codeception

CodeceptJS

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

Getting started

Configuration

Reference

Mocha

To specify a test's location in the [behavior-based hierarchy](#):

Annotations API

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.epic("Web interface");
        Allure.feature("Essential features");
        Allure.story("Authentication");
        // ...
    }
}
```

To specify a test's location in the [suite-based hierarchy](#):

Runtime API

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.label("parentSuite" "Tests for web interface");
        Allure.suite("Tests for essential features");
        Allure.label("subSuite" "Tests for authentication");
    }
}
```

[Run tests](#)

[Generate a report](#)

[Writing tests](#)

[Specify description, links ...](#)

[Organize tests](#)

[Divide a test into steps](#)

[Describe parametrized te...](#)

[Attach screenshots and o...](#)

[Select tests via a test pla...](#)

[Environment information](#)

Introduction

Install & Upgrade

Features

>

Guides

Frameworks

Behat

Behave

Codeception

CodeceptJS

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

Getting started

Configuration

Reference

Mocha

allurereport.org/docs/junit5/#organize-tests

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

To specify a test's location in the [behavior-based hierarchy](#):

Behavior: Epic / Feature / Story

```
Annotations API Runtime API  
import io.qameta.allure.Allure;  
import org.junit.jupiter.api.Test;
```

```
class TestMyWebsite {  
  
    @Test  
    void testAuthentication() {  
        Allure.epic("Web interface");  
        Allure.feature("Essential features");  
        Allure.story("Authentication");  
        // ...  
    }  
}
```

To specify a test's location in the [suite-based hierarchy](#):

```
java  
import io.qameta.allure.Allure;  
import org.junit.jupiter.api.Test;  
  
class TestMyWebsite {  
  
    @Test  
    void testAuthentication() {  
        Allure.label("parentSuite" "Tests for web interface");  
        Allure.suite("Tests for essential features");  
        Allure.label("subSuite" "Tests for authentication");  
    }  
}
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

Install & Upgrade

Organize tests

On this page

Setting up

Add Allure dependencies

Configure JUnit

Configure Results ...

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

allurereport.org/docs/junit5/#organize-tests

Guides

How it works

Frameworks

Behat

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

To specify a test's location in the [behavior-based hierarchy](#):

Annotations API Runtime API

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.epic("Web interface");
        Allure.feature("Essential features");
        Allure.story("Authentication");
        // ...
    }
}
```

To specify a test's location in the [suite-based hierarchy](#):

java

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.label("parentSuite" "Tests for web interface");
        Allure.suite("Tests for essential features");
        Allure.label("subSuite" "Tests for authentication");
    }
}
```

Behavior: Epic / Feature / Story

Runtime API

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

Introduction

Install & Upgrade

>

Organize tests

On this page

Setting up

Add Allure dependencies

Configure JUnit

Configure Results ...

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the

allurereport.org/docs/junit5/#organize-tests

Guides

>

How it works

>

Annotations API Runtime API

import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

```
class TestMyWebsite {  
  
    @Test  
    void testAuthentication() {  
        Allure.epic("Web interface");  
        Allure.feature("Essential features");  
        Allure.story("Authentication");  
        // ...  
    }  
}
```

To specify a test's location in the [behavior-based hierarchy](#):

Behavior: Epic / Feature / Story

Runtime API

Allure.epic("Web interface");

To specify a test's location in the [suite-based hierarchy](#):

Allure.feature("Essential features");

Allure.story("Authentication");

Getting started

Configuration

Reference

Mocha

>

```
@Test  
void testAuthentication() {  
    Allure.label("parentSuite" "Tests for web interface");  
    Allure.suite("Tests for essential features");  
    Allure.label("subSuite" "Tests for authentication");  
}
```

Introduction

[Install & Upgrade](#) >

[Getting started](#) >

[Features](#) >

[Guides](#) >

[How it works](#) >

[Integrations](#) >

[Frameworks](#) >

Behat >

Behave >

Codeception >

CodeceptJS >

Cucumber.js >

Cucumber-JVM >

Cucumber.rb >

Cypress >

Jasmine >

JBehave >

Jest >

JUnit 4 >

JUnit 5 >

[Getting started](#)

Configuration

Reference

Mocha >

Organize tests

As described in [Improving navigation in your test report](#), Allure supports multiple ways to organize tests into hierarchical structures. Allure JUnit 5 provides functions to assign the relevant fields to tests either by adding annotations or "dynamically" (same as for the [metadata fields](#)).

To specify a test's location in the [behavior-based hierarchy](#):

Annotations API Runtime API

```
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.epic("Web interface");
        Allure.feature("Essential features");
        Allure.story("Authentication");
        // ...
    }
}
```



To specify a test's location in the [suite-based hierarchy](#):

```
java
import io.qameta.allure.Allure;
import org.junit.jupiter.api.Test;

class TestMyWebsite {

    @Test
    void testAuthentication() {
        Allure.label("parentSuite" "Tests for web interface");
        Allure.suite("Tests for essential features");
        Allure.label("subSuite" "Tests for authentication");
    }
}
```

On this page

Setting up

Add Allure dependencies

Configure AspectJ

Specifying Allure Results ...

Run tests

Generate a report

Writing tests

Specify description, links ...

Organize tests

Divide a test into steps

Describe parametrized te...

Attach screenshots and o...

Select tests via a test pla...

Environment information

[Introduction](#)[On this page >](#)

Inst

allurereport.org/docs/gettingstarted-navigation

Getting started

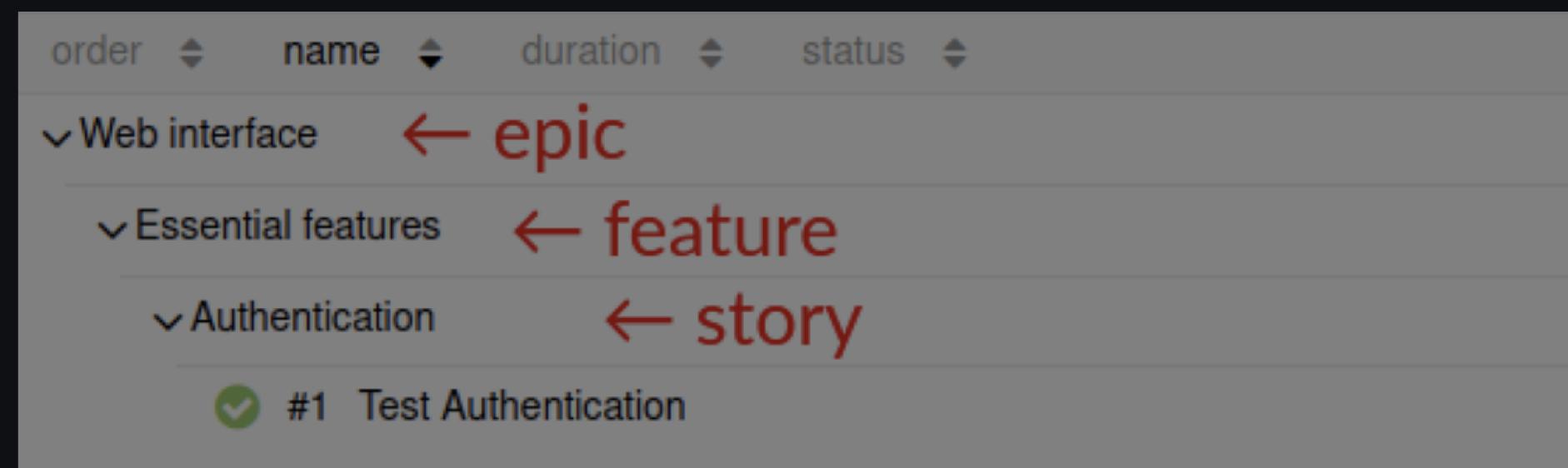
[How to view a report](#)[Improving readability of your test reports](#)[Improving navigation in your test report](#)

Features

[Test steps](#)[Attachments](#)[Test statuses](#)[Sorting and filtering](#)[Defect categories](#)[Visual analytics](#)[Test stability analysis](#)[History and retries](#)

Epics, **features** and **user stories** are widely-used terms for describing a software's features and organizing tests for said features. Usually, an epic is a large set of features that the team aims to develop and test, and a feature is described as a set of user stories that describe how the software is expected to behave in different scenarios.

Allure adapters for all test frameworks provide ways to indicate a test's epic, feature and user story. If a test framework lets the author define some of these terms natively, e.g., in its own language-agnostic file format, then Allure tries its best to incorporate the provided data into the generated tree structure. See the specific Allure adapter's documentation for more details.



Поля по умолчанию

docs.qameta.io/allure-testops/briefly/test-cases/labels/

- parentSuite, suite, subSuite
- testClass, testMethod
- epic, feature, story
- owner, lead
- thread
- layer

Launches

Test Results

Defects

Jobs

Profile

Test cases

Workflow statuses

Manual test scenario

Map

Att

Tags

Layers

Links

Issues

Test keys

Members

Relations

Custom fields

Labels

Import

Launches

Environment

Upload policy

Compare launches

Error categories

Project

Managing access

Cleanup rules

Поля по умолчанию

docs.qameta.io/allure-testops/briefly/test-cases/labels/

- parentSuite, suite, subSuite

- testClass, testMethod

- epic, feature, story

- owner, lead

- thread

- layer

List of standard labels used in Allure Framework

Label	Occurrence*	Allure Report v.2	Allure TestOps	Comment
ALLURE_ID	first	✗	✓	
testClass	first	✓	✓	
testMethod	first	✓	✓	
parentSuite	all	✓	✓	
suite	all	✓	✓	
subSuite	all	✓	✓	
epic	all	✓	✓	
feature	all	✓	✓	
story	all	✓	✓	
framework	first	✓	✗	No default processing for Allure TestOps, but often used in user-defined mappings
language	first	✓	✗	No default processing for Allure TestOps, but often used in user-defined mappings

On this page

Attributes

How it works?

Examples

List of standard labels used in Allure Framework

Задать можно **любые** метки (**label**)

service (сервис)

★ Allure.label("service", "API");

endpoint (метод сервиса)

- ★ Allure.label("service", "API");
- ★ Allure.label("endpoint", "GET /projects");

payload (параметры)

```
Allure.label("service", "API");  
Allure.label("endpoint", "GET /projects");  
★ Allure.label("payload", "fields=item");
```

dataset (тестовые данные)

```
Allure.label("service", "API");  
Allure.label("endpoint", "GET /projects");  
Allure.label("payload", "fields=item");  
★ Allure.label("dataset", "enterprise 10000 users");
```

version (версию сервиса)

```
Allure.label("service", "API");  
Allure.label("endpoint", "GET /projects");  
Allure.label("payload", "fields=item");  
Allure.label("dataset", "enterprise 10000 users");  
★ Allure.label("version", getServerVersion());
```

environment (стенд)

```
Allure.label("service", "API");
Allure.label("endpoint", "GET /projects");
Allure.label("payload", "fields=item");
Allure.label("dataset", "enterprise 10000 users");
Allure.label("version", getServerVersion());
★ Allure.label("environment", "production");
```

server-location (размещение)

```
Allure.label("service", "API");
Allure.label("endpoint", "GET /projects");
Allure.label("payload", "fields=item");
Allure.label("dataset", "enterprise 10000 users");
Allure.label("version", getServerVersion());
Allure.label("environment", "production");
★ Allure.label("server-location", "USA");
```

client-location (параметры клиента)

```
Allure.label("service", "API");
Allure.label("endpoint", "GET /projects");
Allure.label("payload", "fields=item");
Allure.label("dataset", "enterprise 10000 users");
Allure.label("version", getServerVersion());
Allure.label("environment", "production");
Allure.label("server-location", "USA");
★ Allure.label("client-location", "Europe");
```

Задать можно любые метки (label)

```
Allure.label("service", "API");
Allure.label("endpoint", "GET /projects");
Allure.label("payload", "fields=item");
Allure.label("dataset", "enterprise 10000 users");
Allure.label("version", getServerVersion());
Allure.label("environment", "production");
Allure.label("server-location", "USA");
Allure.label("client-location", "Europe");
```



В **Annotation API**

значения всегда **задаются в коде**

```
@Endpoint("GET /projects");
```

В Runtime API

значения могут быть **константами**

```
Allure.label("version", "1.2.3");
```

В Runtime API

значения могут быть константами

```
Allure.label("version", "1.2.3");
```

и функциями тоже

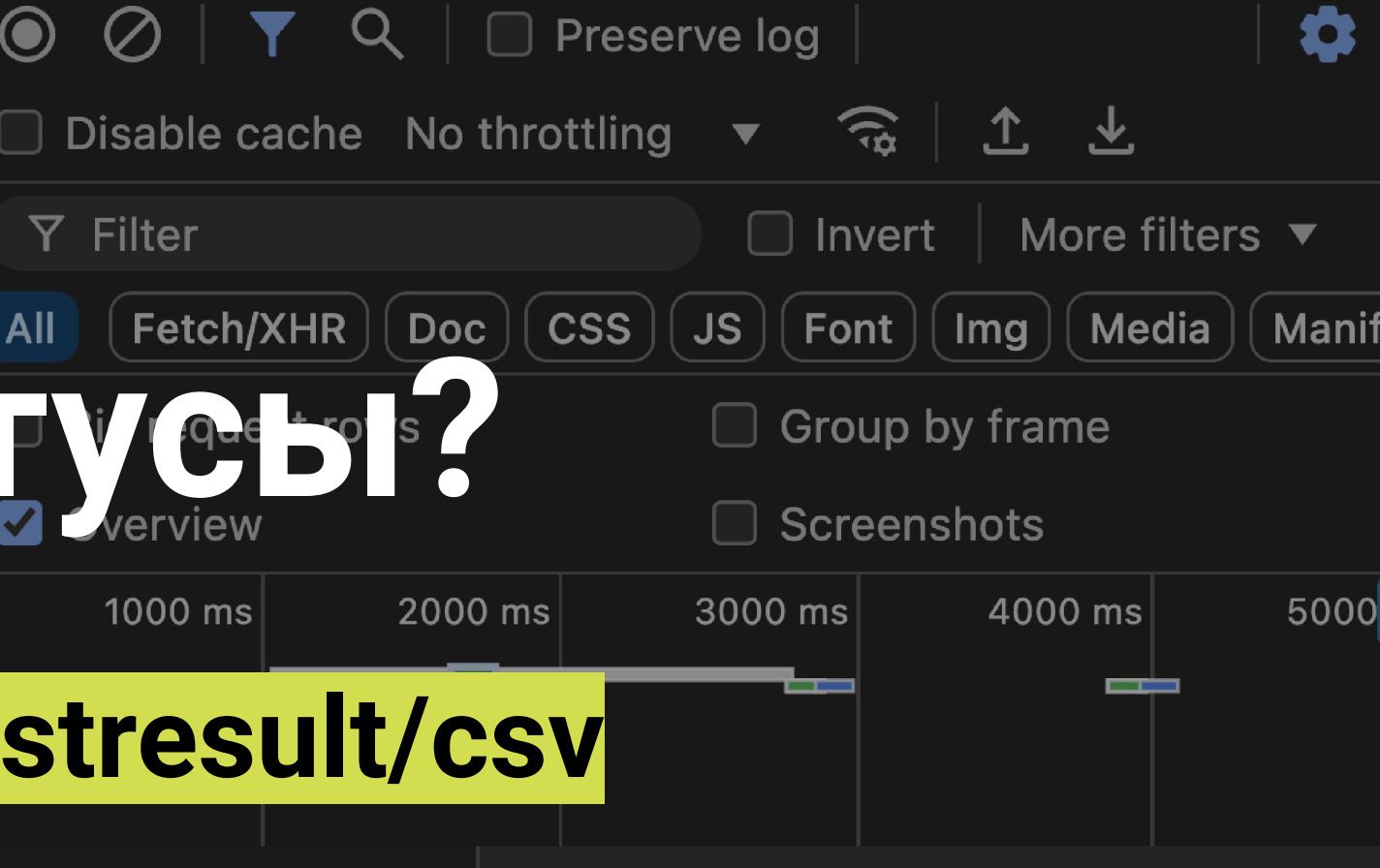
```
Allure.label("version", getServerVersion());
```

Как сгруппировать статусы?

Endpoint	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=name	10	2 8
GET /project	?fields=items	20	20
GET /project	?fields=permissions	20	20

Как сгруппировать статусы?

API Export Launch POST /api/rs/export/testresult/csv



Как сгруппировать статусы?

API Export Launch POST /api/rs/export/testresult/csv

Launch at 2024-11-10 17:40:54

DONE

No en

Copy

Link to an issue

Export to PDF

Export to CSV

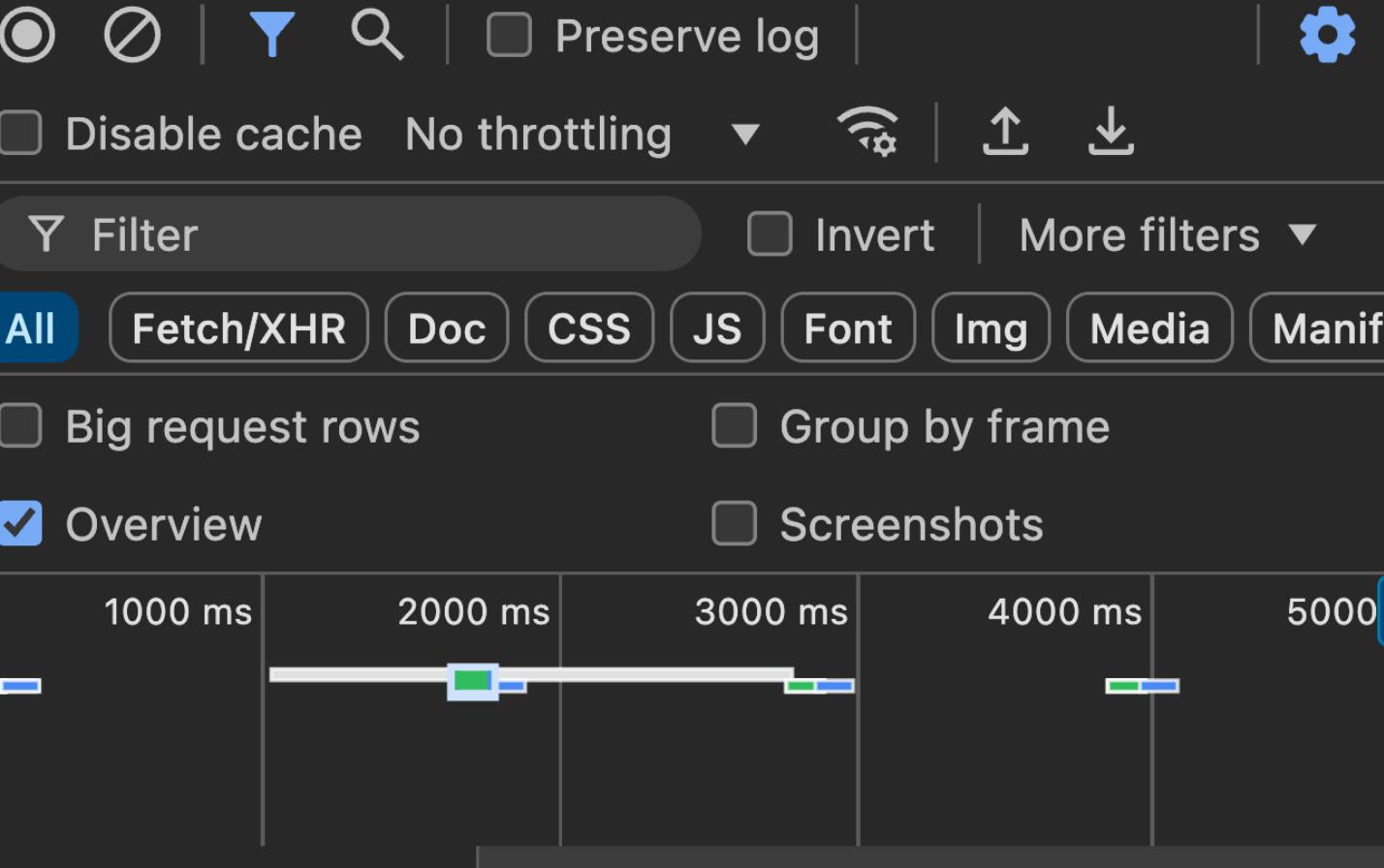
Rerun job

Apply Defect matchers

Edit

Delete

Name	Headers	Payload	»
mapping			▼ Request Payload view source
role			▶ {mapping: [{field: "allure_id", columnSeparator: ";"}, includeHeaders: true}
cf?projectId=62...			▶ mapping: [{field: "allure_id", name: "Launch at 2024-11-10 17:40:54"}]
webclient-infield...			▶ selection: {inverted: true, group: "Launch at 2024-11-10 17:40:54"}
csv			
489			
launch?search=&...			



Name	Headers	Payload	>>
▼ Request Payload view source			
mapping			
role			
cf?projectId=62...			
webclient-infield...			
csv			
489			
launch?search=&...			

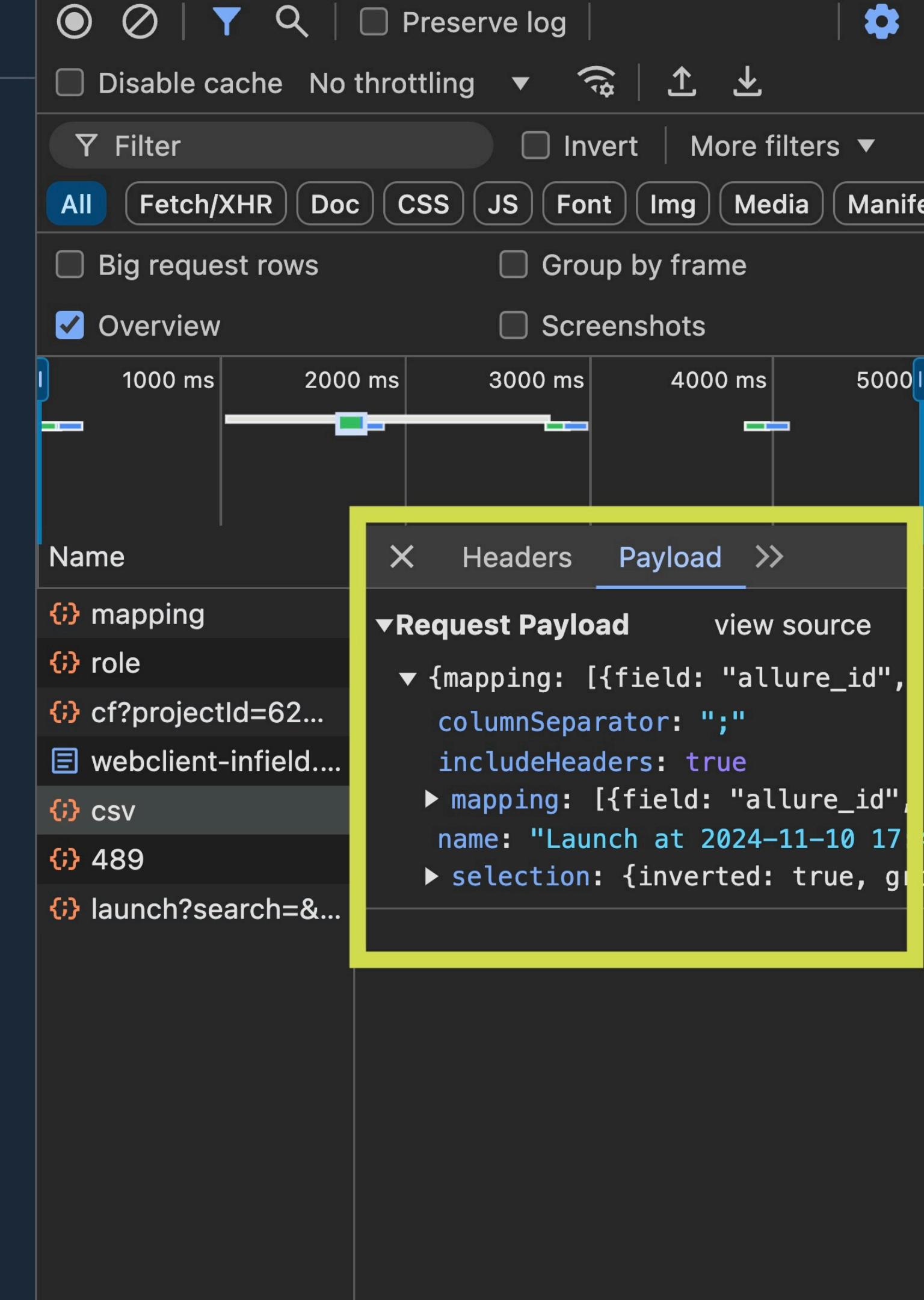
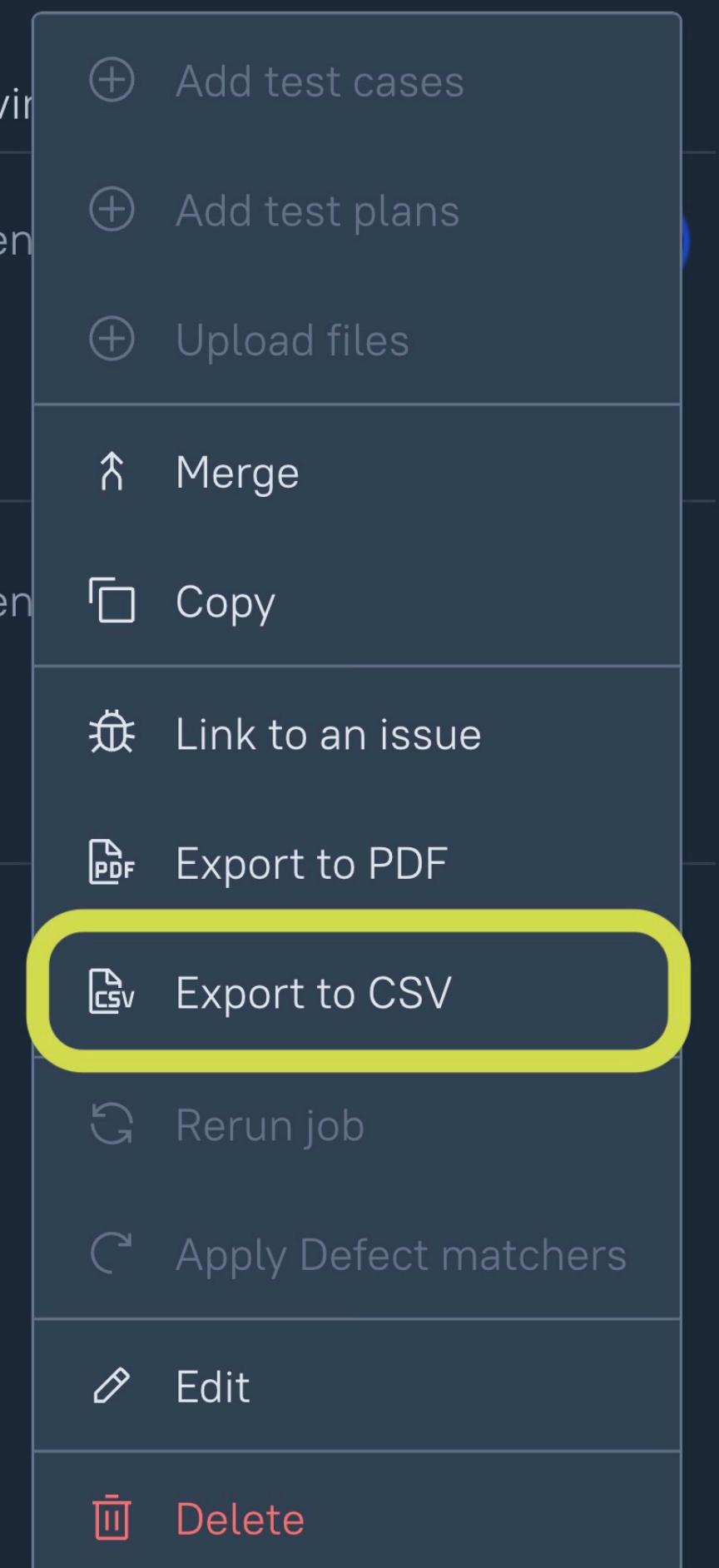
```
▼ {mapping: [{field: "allure_id", columnSeparator: ";"}, includeHeaders: true} ▶ mapping: [{field: "allure_id", name: "Launch at 2024-11-10 17:46:54"}] ▶ selection: {inverted: true, group: "Launch at 2024-11-10 17:46:54"}]
```



• • •



ID	Details	Envir
<input type="checkbox"/> #1326059	<u>Launch at 2024-11-10 17:46:54</u> DONE	No en
<input type="checkbox"/> #1326057	<u>Launch at 2024-11-10 16:57:10</u> DONE	No en



Как сгруппировать статусы?

API Export Launch POST /api/rs/export/testresult/csv

Launch at 2024-11-10 17:40:54

DONE No en

Copy

Link to an issue

Export to PDF

Export to CSV

Rerun job

Apply Defect matchers

Edit

Delete

Name

mapping

role

cf?projectId=62...

webclient-infield...

csv

489

launch?search=&...

Headers

Payload

>>

Request Payload

view source

```
▼ {mapping: [{field: "allure_id",  
    columnSeparator: ";"},  
    includeHeaders: true  
  },  
  mapping: [{field: "allure_id",  
    name: "Launch at 2024-11-10 17:40:54"},  
  ],  
  selection: {inverted: true, g...
```

Как **сгруппировать** статусы?

API Export Launch POST /api/rs/export/testresult/csv

CSV загружается в **InfluxDB** / **VictoriaMetrics** / ...

Как **сгруппировать** статусы?

API Export Launch POST /api/rs/export/testresult/csv

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит **аналитический** отчет

Как **сгруппировать** статусы?

API Export Launch POST /api/rs/export/testresult/csv

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит аналитический отчет

Grafana показывает **детали**

Как сгруппировать статусы?

API Export Launch POST /api/rs/export/testresult/csv

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит аналитический отчет

Grafana показывает детали

Анализ обычного отчета: **2+ часа**

Анализ обычного отчета: 2+ часа

Анализ аналитики: от 10-минут

Анализ обычного отчета: 2+ часа

Анализ аналитики: от 10-минут

Выгодно с 12-го теста

Таблицы в Grafana

дают читателю
легкость анализа

А также в Allure TestOps легко

А также в Allure TestOps легко
создать **дефект** по **Fail**-тесту

А также в Allure TestOps легко

создать дефект по Fail-тесту

и за-Mute-ить (**игнорировать**) сбой

А также в Allure TestOps легко

создать **дефект** по **Fail**-тесту

и за-**Mute**-ить (**игнорировать**) сбой

Читатели

не игнорируют

тесты по нагрузке

Читатели

правильно

игнорируют

тесты по нагрузке

Читатели

правильно (Issue)

игнорируют (Mute)



Читателям

нравится

простота

5 Что дает JUnit писателю?

Allure интегрирован с JUnit 5

- работает хорошо
- документация
- примеры

В **JUnit** понятная **структура** проекта

В JUnit понятная структура проекта

@BeforeAll для подачи **Нагрузки**

- Запуск инструмента (**k6/gatling/jmeter/...**)
- Ожидание метрик (**summary.json**)

В JUnit понятная структура проекта

@BeforeAll для подачи нагрузки

- Запуск инструмента (k6/gatling/jmeter/...)
- Ожидание метрик (summary.json)

@Test для проверок

- Проверка метрик (summary.json)
- Добавление деталей (ссылки на Grafana)

В JUnit понятная структура проекта

@BeforeAll

- Запуск инструмента
- Ожидание метрик

@Test

- Проверка метрик
- Добавление деталей

JUnit написан на **Java**

JUnit написан на Java

Некоторые инструменты запустить легко

- JMeter-Java-DSL
- Gatling

JUnit написан на Java

Некоторые инструменты запустить легко

- JMeter-Java-DSL
- Gatling

Любые инструменты запустить возможно

- В Java есть [java.testcontainers.org](https://java-testcontainers.github.io/)
- Через Test Containers можно запустить Docker-контейнеры
- Внутри Docker можно запустить: k6, Taurus, ...

JUnit может запустить

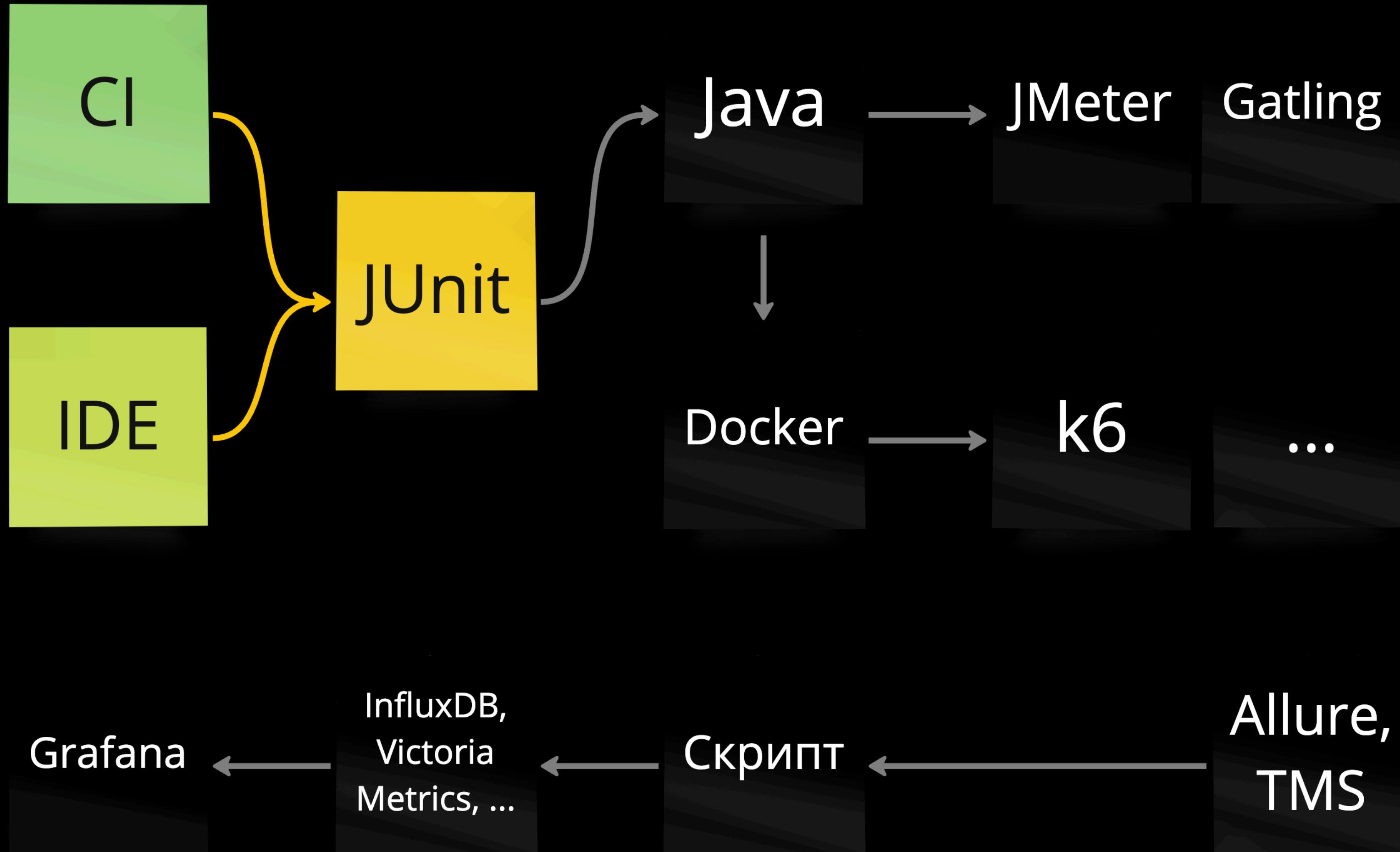
- JMeter-Java-DSL
- Gatling
- Taurus
- k6
- любой инструмент

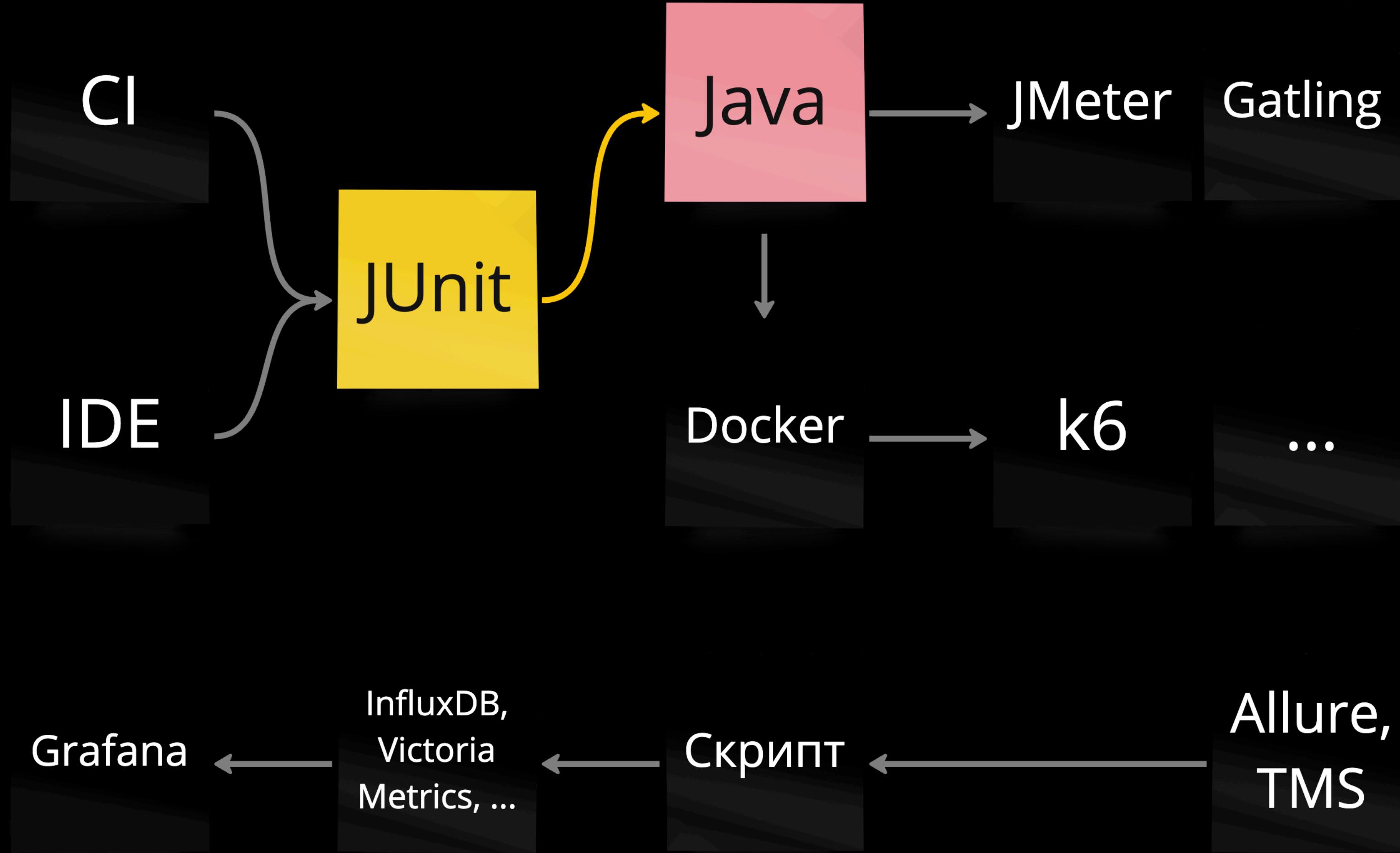


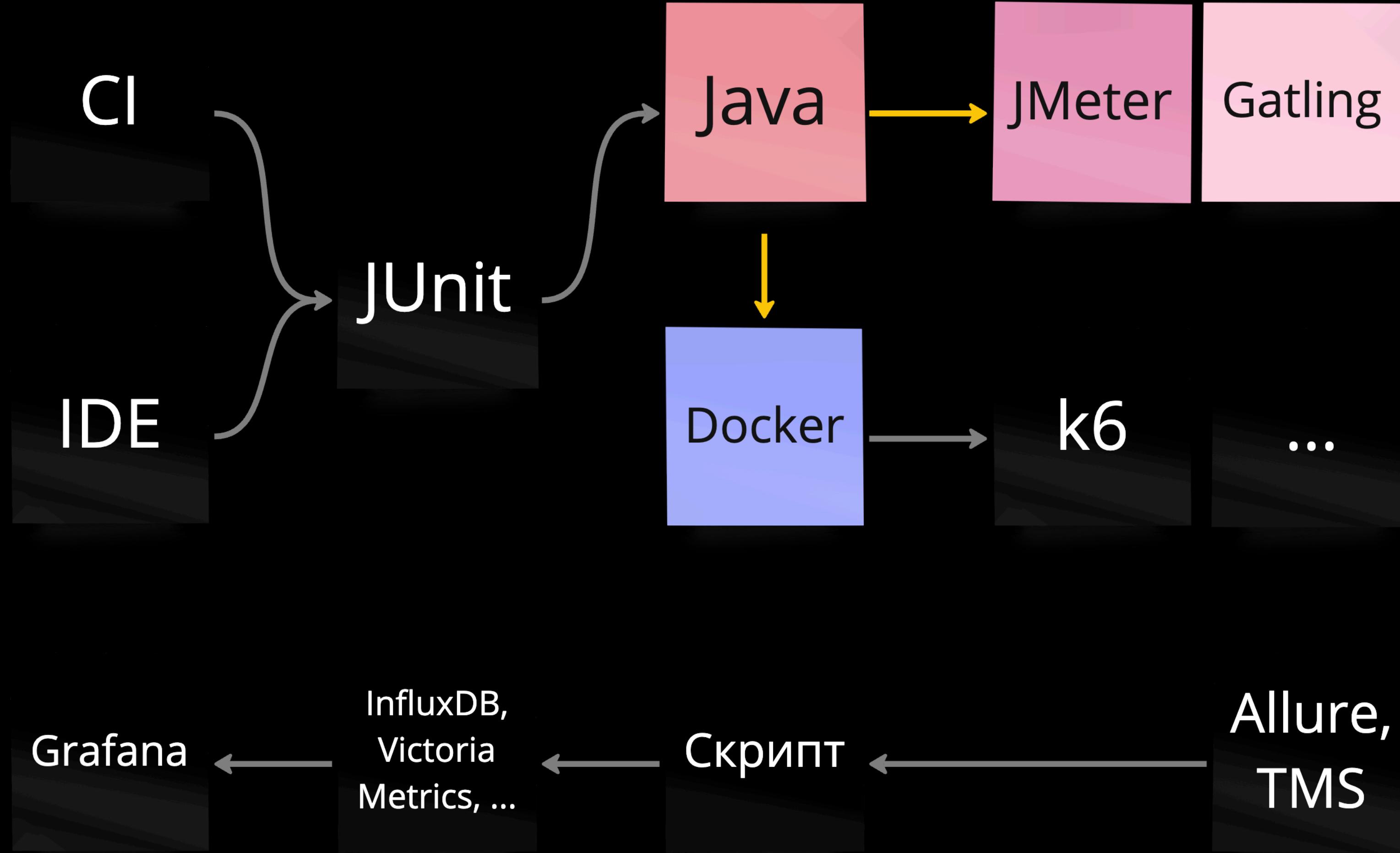
интегрирован в
среды разработки IDE

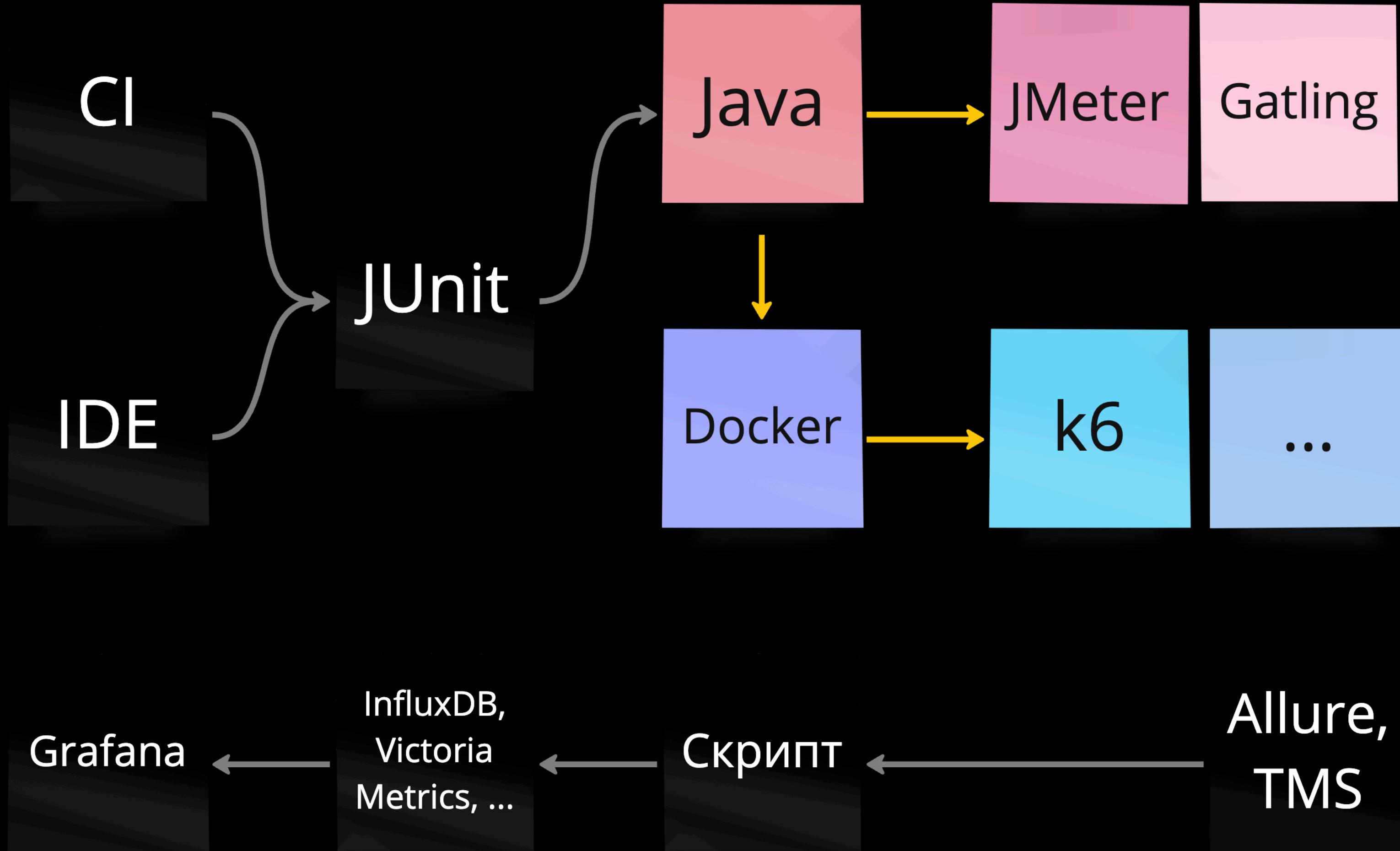


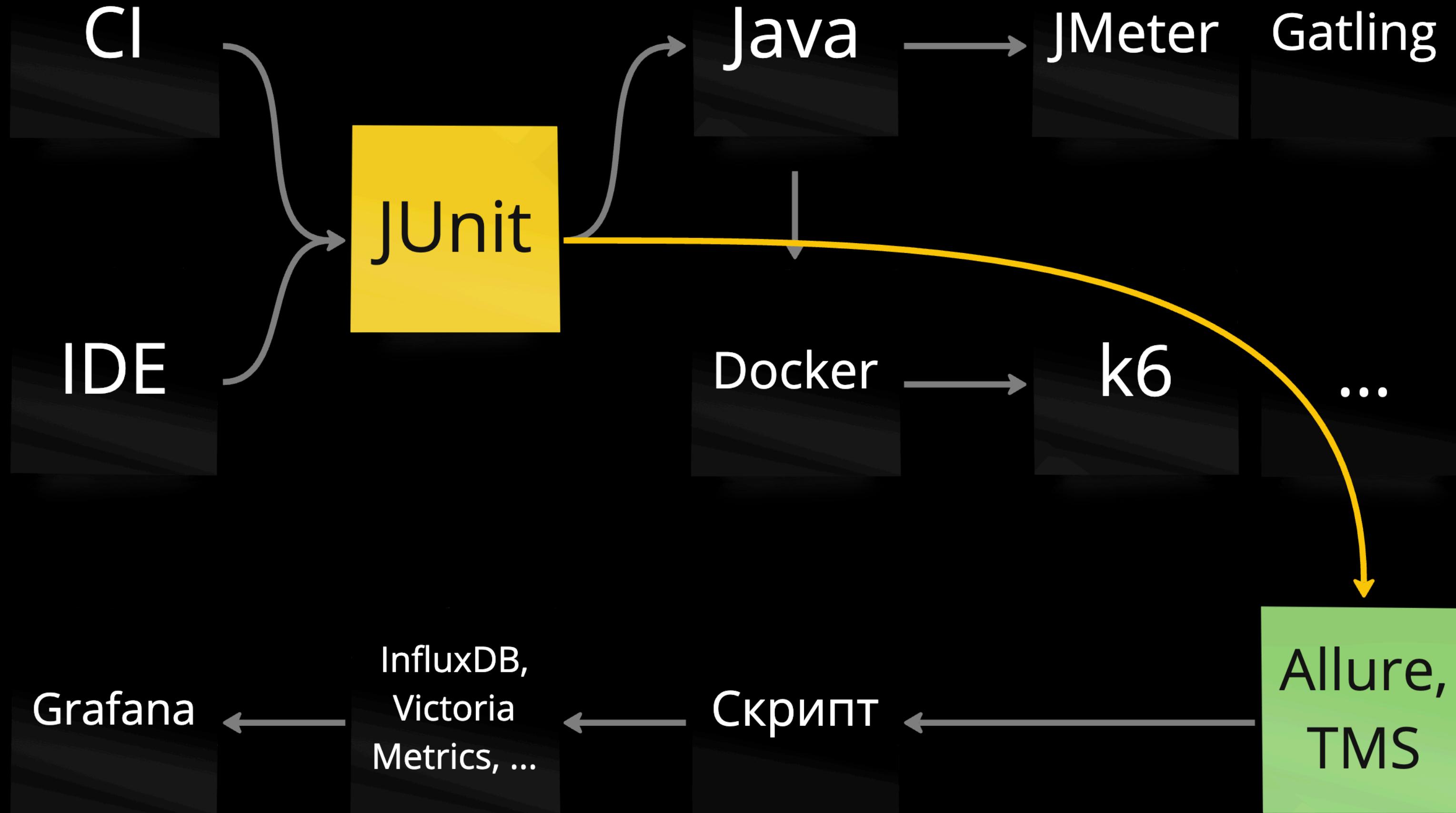
интегрирован в
системы сборки CI

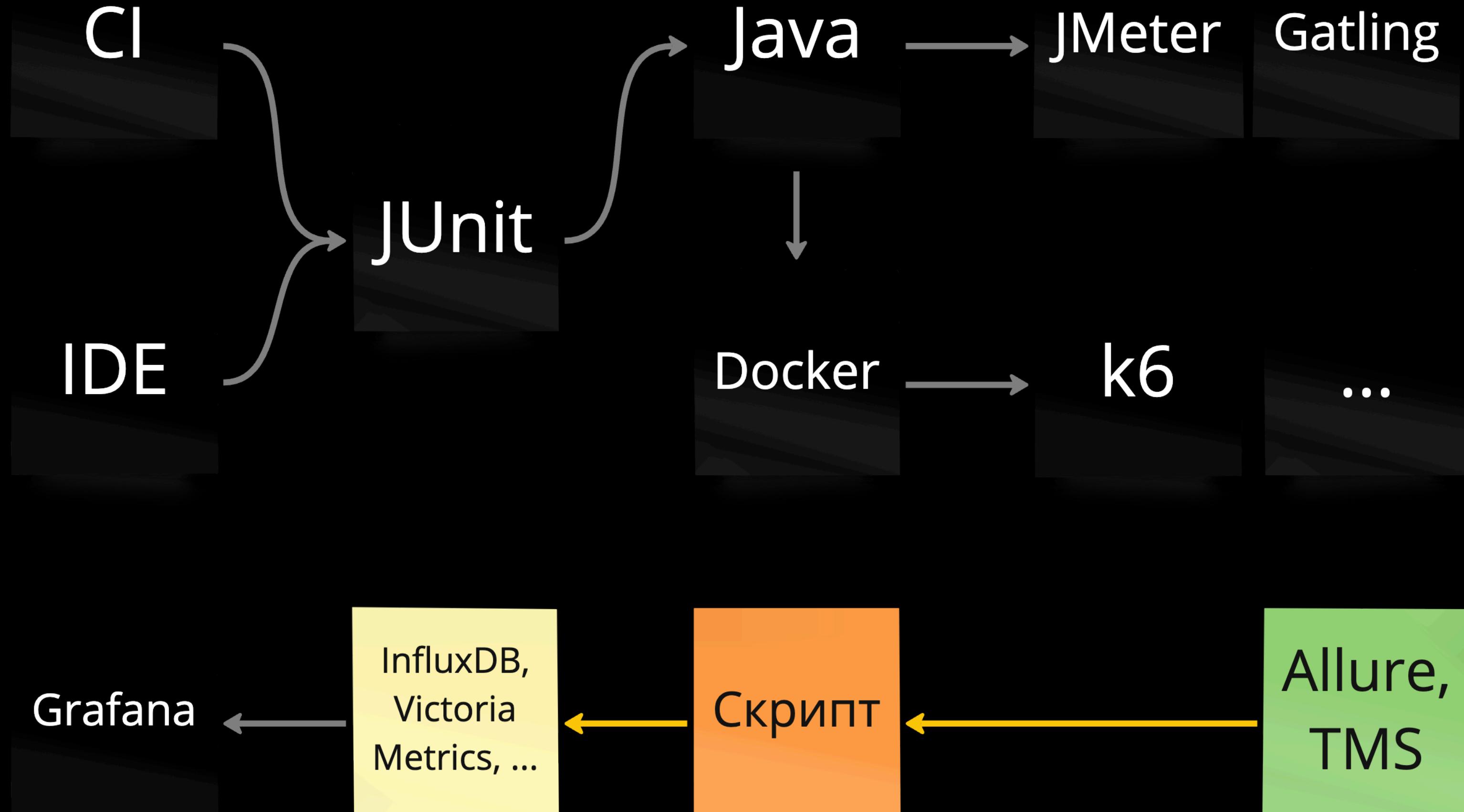


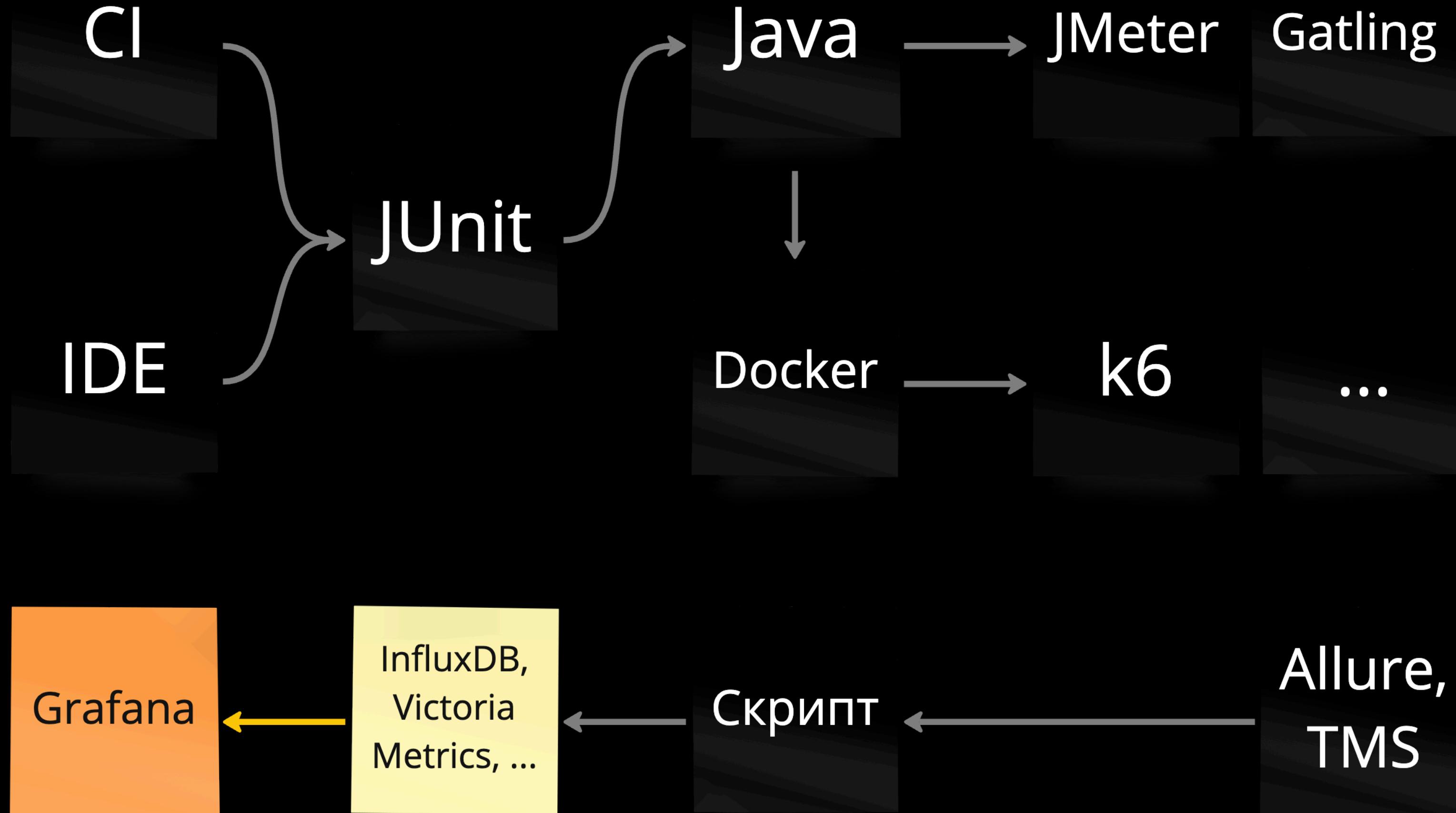


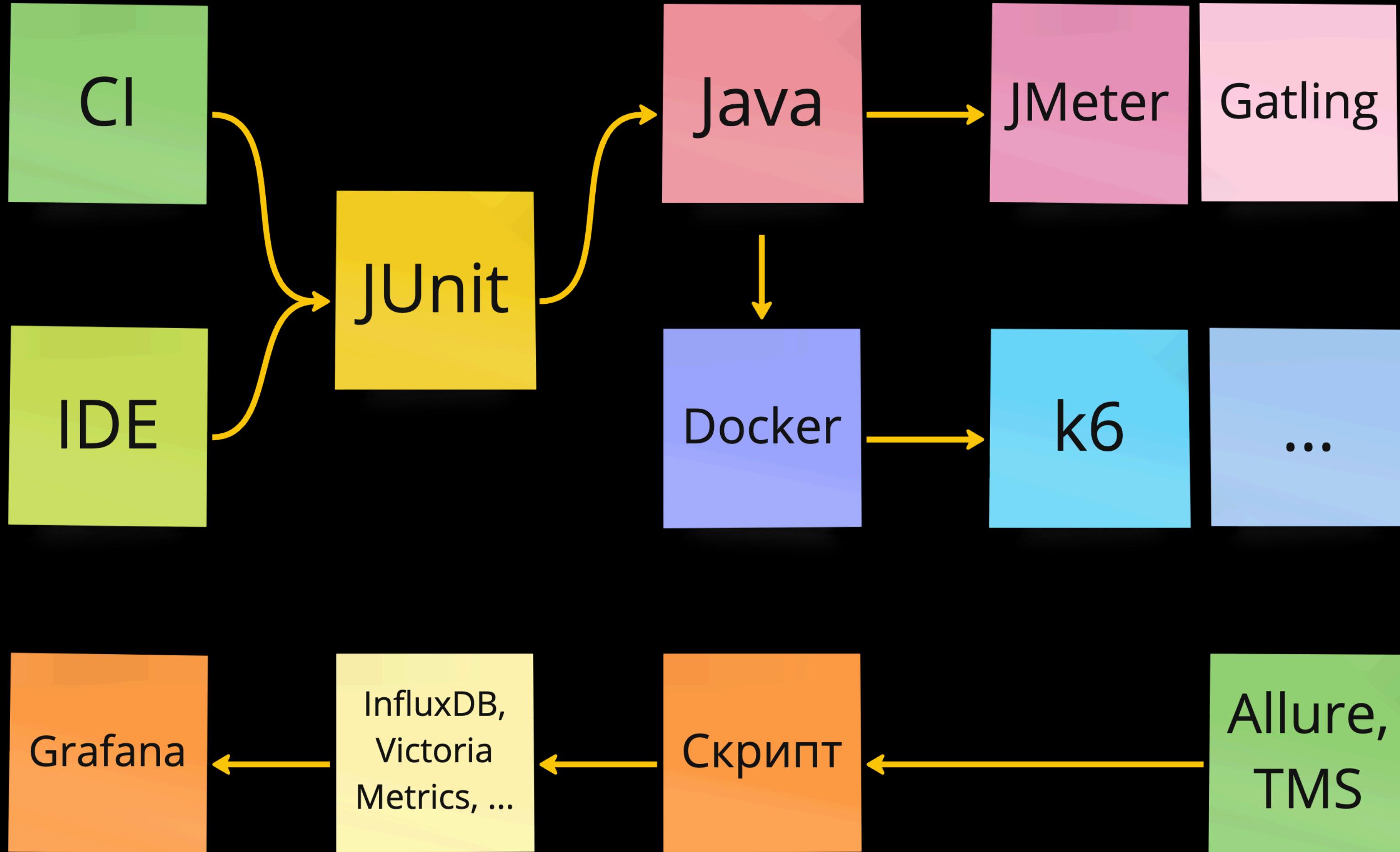












Фреймворк задает структуру

JUnit задает структуру

JUnit задает структуру

Статистика тестируется

JUnit задает структуру

к б **summary.json** тестируется

JUnit задает структуру

к б summary.json тестируется

Шлем результаты в хранилище

JUnit задает структуру

кб summary.json тестируется

Шлем CSV в InfluxDB

JUnit задает структуру

k6 summary.json тестируется

Шлем CSV в InfluxDB

Grafana строит отчет

JUnit задает структуру

кб summary.json тестируется

Шлем CSV в InfluxDB

Grafana строит отчет

6

Почему **больше** тестов – лучше?

Много тестов

увеличивают

тестовое покрытие

Много тестов

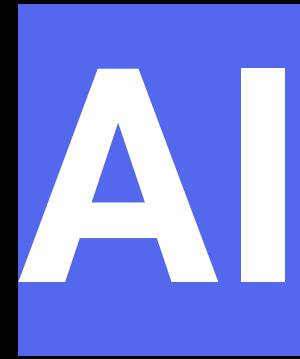
стандартизируют

подачу нагрузки

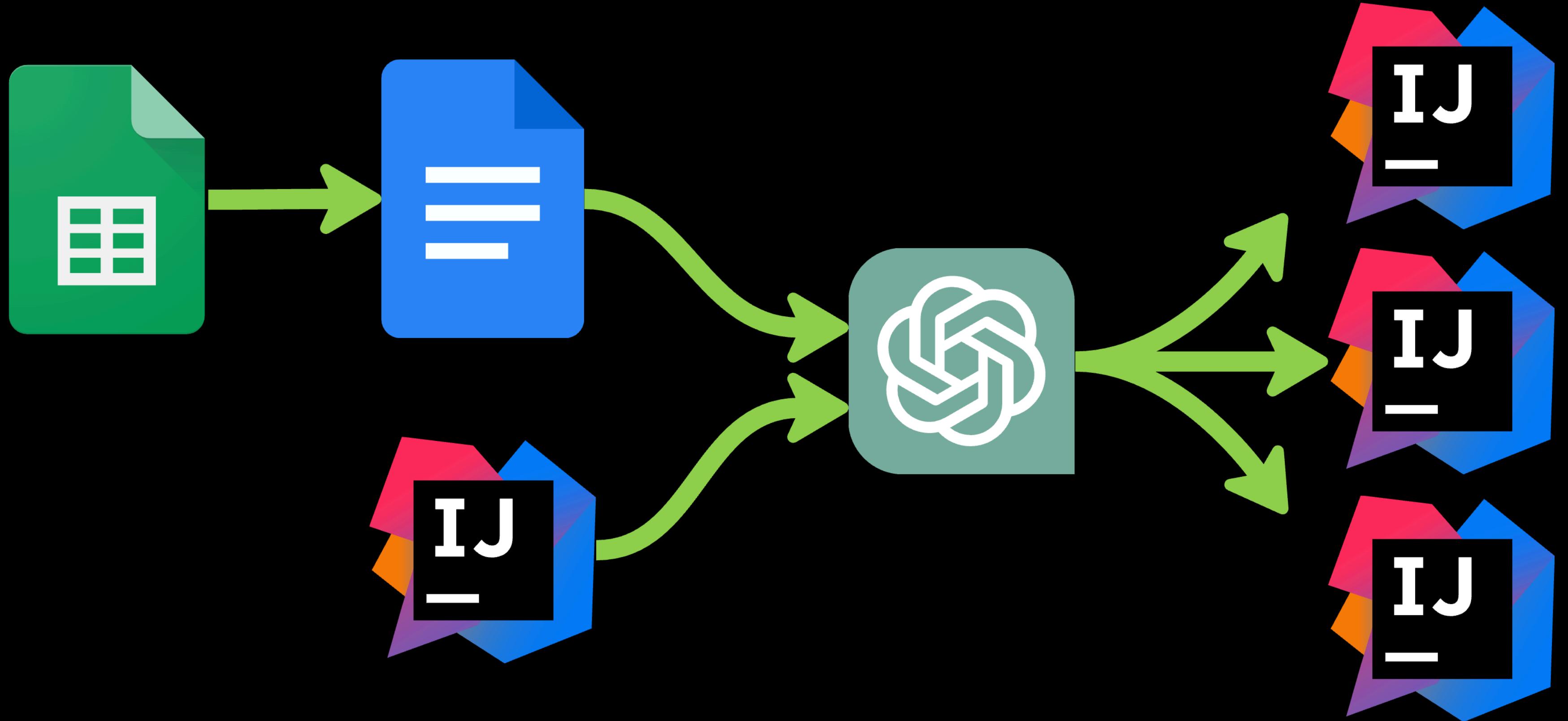
Шаблонные тесты

генерируются

в 2024-м году



легко **генерирует** код
из списка параметров

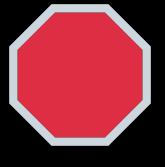




Сотня простых тестов

долго выполняется,

да быстро понимается

Понравилось?  OK  Fail

[@qa_load](#)

[@smirnovqa](#)

[github.com/polarnik/k6-junit-integration](#)

[polarnik.github.io/k6-junit-integration/codetalks](#)

