

Как **количество** тестов  
производительности  
переходит в **качество**

Ночь ...



**Ночь**

**вдруг просыпается нагрузчик со  
словами: "Отчет не отправил!"**

# Автотести

# **АВТОТЕСТЫ**

## **производительности.**

**Автотесты**

производительности,

которых **много**

Автотесты

производительности,

которых много

и это хорошо

# Смирнов Вячеслав

Ускоряю [miro.com](#)

Развиваю [@qa\\_load](#)

# План

1. В чем **проблема** тестирования производительности?
2. Как понять **результат** теста за секунду?
3. Как понять **сотню** **результатов** тестов за минуту?
4. Что дает **Allure** читателю?
5. Что дает **JUnit** писателю?
6. Почему **больше** тестов – лучше?
7. Какие есть **примеры**?

1 В чем **проблема**  
тестирования  
производительности?

Просто ли написать тесты  
производительности?

# Непростые

## тесты

Просто ли **понять отчет** по  
производительности?

Непростые тесты

и

непростые результаты

# Непростые тесты

и

непростые результаты

## 2 Как понять результат теста за секунду?

Простейший результат **теста**:

 хорошо

 плохо

Простейший результат **теста**:

 хорошо

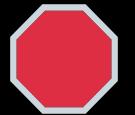
 плохо

 не получилось проверить

# Результат набора тестов:

 15

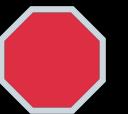
 10  5

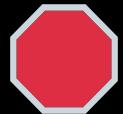
 15

 15

# Результат набора тестов:

 15 – все хорошо

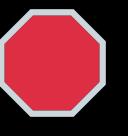
 10  5

 15

 15

# Результат набора тестов:

 15

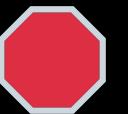
 10  5 – **кое-что сломалось**

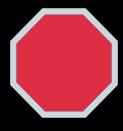
 15

 15

# Результат набора тестов:

 15

 10  5

 15 – **все плохо**

 15

# Результат набора тестов:

 15

 10  5

 15

 15 – не удалось проверить

# 3 Как понять сотню результатов тестов за минуту?

# Что с API?

API	Version 1.2.3	Version 1.2.4
GET /users	80	80
POST /project	60	60
GET /project	60	52  8
GET /account	50	50

# Сбой GET /project для версии 1.2.4

API	Version 1.2.3	Version 1.2.4
GET /users	✓ 80	✓ 80
POST /project	✓ 60	✓ 60
<b>GET /project</b>	✓ 60	✓ 52 ⚡ 8
GET /account	✓ 50	✓ 50

# ЧТО НЕ ТАК С GET /project?

API	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=id	✓ 15	✓ 15
GET /project	?fields=name	✓ 15	✓ 15
GET /project	?fields=items	✓ 15	✓ 7 ⚡ 8
GET /project	?fields=permissions	✓ 15	✓ 15

# Поле **items** замедлилось в 1.2.4

API	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=id	✓ 15	✓ 15
GET /project	?fields=name	✓ 15	✓ 15
GET /project	?fields=items	✓ 15	✓ 7 ⚡ 8
GET /project	?fields=permissions	✓ 15	✓ 15

**Аналитический отчет или  
сводная таблица  
позволяют быстро оценить**

4

# Что дает Allure читателю?

## 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

Dependencies

[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](#)

# https://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    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");
    }
}
```

## 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](#)

dependencies

[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](#)

# <https://allurereport.org/docs/junit5/#organize-tests>

Guides

# Behavior: Epic / Feature / Story

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](#):

```
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");
    }
}
```

## Organize tests

### On this page

Setting up

dencies

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

# <https://allurereport.org/docs/junit5/#organize-tests>

Guides

How to ...

Integrations

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");
    }
}
```

# Annotations API

Codeception

CodeceptJS

Cucumber.js

Cucumber-JVM

Cucumber.rb

Cypress

Jasmine

JBehave

Jest

JUnit 4

JUnit 5

Getting started

Configuration

Reference

Mocha

**Organize tests****On this page**

Setting up

Dependencies

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

# <https://allurereport.org/docs/junit5/#organize-tests>

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

# Behavior: Epic / Feature / Story

## Annotations API

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

class TestWebsite {

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

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");
        // ...
    }
}
```

## 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

Dependencies

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

# https://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    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");
    }
}
```

## 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

Dependencies

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

# <https://allurereport.org/docs/junit5/#organize-tests>

Guides

# Behavior: Epic / Feature / Story

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 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");
    }
}
```

## Organize tests

### On this page

[Setting up](#)

dencies

[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](#)

# <https://allurereport.org/docs/junit5/#organize-tests>

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

# Behavior: Epic / Feature / Story

```
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");
    }
}
```

# Runtime API

- Codeception >
- CodeceptJS >
- Cucumber.js >
- Cucumber-JVM >
- Cucumber.rb >
- Cypress >
- Jasmine >
- JBehave >
- Jest >
- JUnit 4 >
- JUnit 5 <div data-bbox="200 840 215 855" style="float: right;">▼

### Getting started

[Configuration](#)[Reference](#)

- Mocha >

**Organize tests****On this page**

Setting up

Dependencies

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

**<https://allurereport.org/docs/junit5/#organize-tests>**To specify a test's location in the [behavior-based hierarchy](#):

# Behavior: Epic / Feature / Story

## Runtime API

```
Allure.epic("Web interface");
Allure.feature("Essential features");
Allure.story("Authentication");
```

```
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");
    }
}
```

```
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");
    }
}
```



java

## 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

Get

How

# <https://allurereport.org/docs/gettingstarted-navigation/#behavior-based-hierarchy>

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

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.

order	name	duration	status
1	Web interface	← epic	
2	Essential features	← feature	
3	Authentication	← story	
4	#1 Test Authentication		

# Поля по умолчанию

<https://docs.qameta.io/allure-testops/briefly/test-cases/labels/>

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

## List of standard labels used in Allure Framework

Label	Occurrence*	Allure Report v.2	Allure TestOps	Comment
ALLURE_ID	first	✗	✓	
AS_ID	first	✗	✓	Deprecated, use ALLURE_ID instead
package	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

# Поля по умолчанию

On this page  
Attributes  
How it works?  
Examples  
List of standard labels used in Allure Framework

Workflow statuses

Labels  
Layers

Attachments

Tags

- parentSuite, suite, subSuite

Links

Issues

- testClass, testMethod

Test keys

Members

Relations

- epic, feature, story

Custom fields

Labels

- owner, lead

Launches

Environment

Upload policy

Compare labels

Error categories

Project

- host

Managing access

Cleanup rules

Задать можно любые метки (label)

# Задать можно service (сервис)

★ Allure.label("service", "API");

# **endpoint** (метод сервиса)

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

# **payload** (параметры метода сервиса)

```
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** (тестируемое окружение)

```
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 (параметры окружения)

```
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");
```



...

Значения могут быть **константами**

в **Annotation API**

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

Значения могут быть **константами**

в **Runtime API**

```
Allure.label("endpoint", "GET /projects");
```

Значения могут быть **функциями**

в **Runtime API**

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

# Как построить аналитический отчет?

API	Payload	Version 1.2.3	Version 1.2.4
GET /project	?fields=id	✓ 15	✓ 15
GET /project	?fields=name	✓ 15	✓ 15
GET /project	?fields=items	✓ 15	✓ 7 ⚡ 8
GET /project	?fields=permissions	✓ 15	✓ 15

# Как построить аналитический отчет?

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

The screenshot shows the Allure Performance Tests interface. At the top, there are navigation icons for 'PE' (Performance Tests), 'Launches', and 'Reports'. The main area displays a 'Launch' card with ID #1326059, created at 2024-11-10 17:46:54. The card includes sections for 'Details' (Launch at 2024-11-10 17:46:54, Done), 'Env' (Add test cases, Add environment, Upload files), and a 'Logs' section. On the right, a 'Logs' panel shows a timeline from 1000 ms to 5000 ms with several entries. A 'Filter' bar at the top right allows selecting 'All' or specific log types like 'Fetch/XHR', 'Doc', 'CSS', 'JS', 'Font', 'Img', 'Media', and 'Manifest'. Below the logs, there are sections for 'Overview' and 'Screenshots'. A large yellow banner at the bottom of the page reads 'API Export Launch POST /api/rs/export/testresult/csv'.

# Как построить аналитический отчет?

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

Launch at 2024-11-10 17:46:54 **DONE**  
No entries  
#1326059 Created 2024-11-10 16:57:10

- Copy
- Link to an issue
- Export to PDF
- Export to CSV
- Rerun job
- Apply Defect matchers
- Edit
- Delete

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

Preserve log | 

Disable cache No throttling   

 Filter  Invert More filters 

All Fetch/XHR Doc CSS JS Font Img Media Manifest

Big request rows  Group by frame

Overview  Screenshots

1000 ms 2000 ms 3000 ms 4000 ms 5000 ms

Name	Headers	Payload	»
<b>▼ Request Payload</b> <a href="#">view source</a>			
mapping			
role			
cf?projectId=62...			
webclient-infield...			
csv			
489			
launch?search=&...			

ID	Details	Envir	Actions
<input type="checkbox"/> #1326059	<a href="#">Launch at 2024-11-10 17:46:54</a> 	No en	<ul style="list-style-type: none"><li> Add test cases</li><li> Add test plans</li><li> Upload files</li><li> Merge</li><li><input type="checkbox"/> Copy</li><li> Link to an issue</li><li> Export to PDF</li><li> <b>Export to CSV</b></li><li> Rerun job</li><li> Apply Defect matchers</li><li> Edit</li><li> Delete</li></ul>
<input type="checkbox"/> #1326057	<a href="#">Launch at 2024-11-10 16:57:10</a> 	No en	



...





...



ID

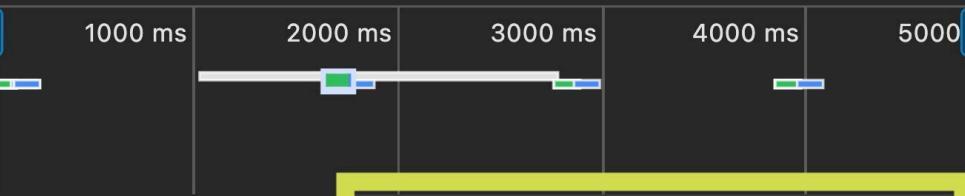
Details

Envir

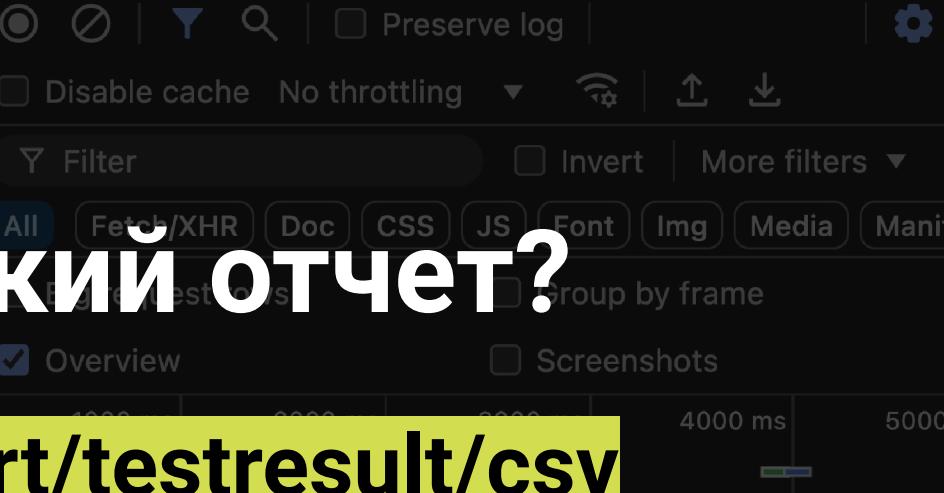
[Launch at 2024-11-10](#) DONE  
Created 2024-11-10  
17:46:54

[Launch at 2024-11-10](#) DONE  
Created 2024-11-10 16:57:10

- Add test cases
- Add test plans
- Upload files
- Merge
- Copy
- Link to an issue
- Export to PDF
- Export to CSV
- Rerun job
- Apply Defect matchers
- Edit
- Delete

 Preserve log |  Disable cache  No throttling Filter  Invert  More filters All Fetch/XHR Doc CSS JS Font Img Media Manifest Big request rows  Group by frame Overview  Screenshots

Name	Headers	Payload	»
<b>▼ Request Payload</b> <a href="#">view source</a>			
mapping			
role			
cf?projectId=62...			
webclient-infield....			
csv			
489			
launch?search=&...			



# Как построить аналитический отчет?

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

The screenshot shows a dark-themed user interface for a performance testing tool. On the left, there's a sidebar with icons for PE, ID, Details, Envir, Add test cases, and Upload files. The main area displays a list of launches. A specific launch entry is selected, showing its ID (#1326059), creation date (Created 2024-11-10 17:46:54), and a status bar indicating 'Launch at 2024-11-10 16:57:10' with a 'DONE' button. A context menu is open over this launch entry, listing options like Copy, Link to an issue, Export to PDF, and Export to CSV. The 'Export to CSV' option is highlighted with a yellow oval. To the right of the launch list, a large modal window is open, showing a detailed view of the request payload for the 'Export to CSV' API call. The payload includes fields for mapping, role, cf?projectId=62..., webclient-infield..., csv, 489, and launch?search=&... . The 'Request Payload' section is expanded, showing the JSON structure of the data being sent to the API.

Launch at 2024-11-10 16:57:10 **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

```
{<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"}]}
```

# Как построить аналитический отчет?

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

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

# Как построить аналитический отчет?

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

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит аналитический отчет

# Как построить аналитический отчет?

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

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит аналитический отчет

Grafana показывает детали

# Как построить аналитический отчет?

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

CSV загружается в InfluxDB / VictoriaMetrics / ...

Grafana строит аналитический отчет

Grafana показывает детали

Анализ обычного отчета: **от 2-х часов**

**Анализ обычного отчета: от 2-х часов**

**Анализ аналитики: от 10-минут**

**Анализ обычного отчета: от 2-х часов**

**Анализ аналитики: от 10-минут**

**Аналитика выгодна уже с 12-го теста**

**Allure TestOps** и **Grafana**  
дают читателю **скорость**  
**анализа**

**А также ...**

А также ...

легко привязать **дефект** или **задачу** к  
результату теста производительности

А также ...

легко привязать **дефект** или **задачу** к  
результату теста производительности

А также ...

легко привязать **дефект** или **задачу** к  
результату теста производительности

и сделать **mite** такому тесту

Читатели **не игнорируют**  
тесты производительности

Читатели **не** игнорируют  
тесты производительности  
более осознанно,  
используя беклог задач

5

# Что дает JUnit писателю?

5

# Что дает JUnit писателю?

# 6 Почему **больше** тестов — лучше?

7

# Какие есть примеры?