



OTUS


ОНЛАЙН ОБРАЗОВАНИЕ

# Онлайн-образование





# Меня хорошо видно && слышно?

Ставьте , если все хорошо  
Напишите в чат, если есть проблемы

Проверить, идет ли запись!





O T U S

ОНЛАЙН ОБРАЗОВАНИЕ

Allure



Андрей Гридяев

Ведущий инженер по автоматизации тестирования

[andrey.gridyaev@gmail.com](mailto:andrey.gridyaev@gmail.com)



## Андрей Гридяев

- Опыт в IT более 10 лет
- Опыт в автоматизации 5 лет
- Языки: Python и Java

# Правила вебинара



Активно участвуем



Задаем вопрос в чат



Off-topic обсуждаем в Slack в канале #general



Вопросы вижу в чате, могу ответить не сразу

# Маршрут вебинара

Обзор Allure



Установка Allure



Работа с Allure



Рефлексия



# Цели вебинара | После занятия вы сможете

1 Установить Allure локально и настроить отчетность в тестах

2 Настроить генерацию Allure-отчетов в Jenkins



# Смысл | Зачем вам это уметь

1 Удобная отчетность по тестам для разработчиков, QA и менеджмента

2 Прозрачность процесса автоматизации тестирования



# Обзор Allure

# Что такое Allure

Allure - инструмент для построения понятных отчетов автотестов

Создание отчета состоит из 2-х этапов

- Во время исполнения теста testing framework adapter собирает данные об исполнении теста и сохраняет их в файлах
- На основании данных сохраненных в файлах происходит генерация HTML-отчета с помощью command line tool, плагина для CI или build tool

Thucydides - фреймворк, на базе которого был сделан Allure

- Java-фреймворк (тесты можно писать только на Java)
- Ориентирован на приемочное тестирование web-приложений
- Имеет монолитную архитектуру



# Почему Allure

- Прозрачная отчетность не только для разработчиков и QA
- Возможность соотнести автотесты с тест-кейсами
- Понимание, какая именно функциональность покрыта тестами
- Минимум времени на разбор отчета
- Интеграция с тестовыми фреймворками (pytest, TestNG и т. д.)
- Поддержка CI (Jenkins, TeamCity)

The background of the image is an aerial view of a city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer that features a network of white lines and dots, resembling a digital or data network. The text "Установка Allure" is centered in the middle of the image in a white, bold, sans-serif font.

# Установка Allure

# Адаптер

**Адаптер** - ПО, которое собирает данные о результатах выполнения тестов

**pytest plugin**

```
$ pip install allure-pytest
```



# Генератор отчетов

**Генератор отчетов** - ПО, которое на основе данных собранных адаптером генерирует Allure-отчет

## **из пакета**

```
$ sudo apt-add-repository ppa:gameta/allure
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install allure
```

## **из архива**

<https://github.com/allure-framework/allure2/releases>

The background of the slide is an aerial photograph of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue and teal gradient. A network of thin, light blue lines connects various points across the gradient, creating a digital or technological feel. The text "Работа с Allure" is centered in the middle of the slide in a white, bold, sans-serif font.

# Работа с Allure

# Структура отчета

**Overview** - общая информация о тестовом запуске

**Categories** - категории дефектов тестов. По умолчанию доступны две категории: *Product defects* (failed tests) и *Test defects* (broken tests)

**Suites** - стандартная группировка результатов тестов по suites и classes

**Graphs** - статистика, например, диаграмма по статусам тестов, график длительности выполнения тестов и т. д.

**Timeline** - диаграмма, показывающая распределение тестов в зависимости от длительности выполнения

**Behaviors** - группировка результатов по тегам Epic, Feature и Story

**Packages** - группировка результатов по package



# Статусы, фикстуры и параметризация

**Passed**

тест успешно пройден

**Failed**

произошло исключение `AssertionError`

**Skipped**

тест пропущен

**Broken**

в тесте произошло исключение (не `AssertionError`)

Отображение задействованных фикстур в секциях **Set up** и **Tear down**

Отображение параметров, с которыми был запущен тест

# Titles, descriptions

Имя теста в отчете можно поменять с помощью декоратора `@allure.title`

`@allure.title` поддерживает вставку аргументов теста и динамическую замену

С помощью декоратора `@allure.description` в тесты можно добавить описание с необходимым уровнем детализации

`@allure.description_html` позволяет передать HTML-документ, который будет отображен в секции **Description**

Если в тесте заполнен docstring - его содержимое будет отображено в секции **Description**

Описание теста может быть динамически изменено с помощью `allure.dynamic.description`

# Steps

Allure позволяет настроить детализированное пошаговое представление теста в виде **steps**

Декоратор **@allure.step** добавляет к отчету вызов декорированного метода или функции с параметрами

Методы декорированные **@allure.step** могут храниться отдельно от тестов и импортироваться при необходимости

Методы декорированные **@allure.step** могут иметь произвольную глубину вложенности



# Метки

В Allure доступны 3 типа меток, с помощью которых можно структурировать представление отчета

- BDD-метки обозначающие Epics, Features и Stories
- Severity-метки
- Custom-метки

# Links

Интеграция с bugtracker-ами или с TMS может осуществляться с помощью **@allure.link**, **@allure.issue**, **@allure.testcase**

Ссылки отображаются в секции **Links**

**@allure.issue** в качестве параметра может передаваться ID бага, который затем будет подставлен в соответствующий шаблон ссылки

Шаблоны ссылок указываются с помощью опции **--allure-link-pattern** и используют типы **issue**, **link** и **test\_case**

Шаблоны и типы ссылок отделяются двоеточием

```
$ pytest tests_dir --alluredir=/tmp/my_allure_report \  
--allure-link-pattern=issue:http://www.bugtracker.com/issue/{}
```

# Attachments

Отчеты могут отображать различные типы вложений, тем самым дополняя результат **test**, **step** или **fixture**

Вложения могут быть добавлены 2-мя способами

- **allure.attach**(body, name, attachment\_type, extension)
- **allure.attach.file**(source, name, attachment\_type, extension)

Вложения отображаются в контексте того теста, которому они принадлежат

**HTML**-вложения рендерятся и отображаются на странице отчета



# Jenkins integration

Как [настроить](#) генерацию Allure-отчетов в Jenkins?

1. Установить [Allure Plugin](#)
2. Выполнить настройку "Allure Commandline" в Global Tool Configuration
3. В настройках сборки добавить в Post-build Actions добавить Allure Report
4. В настройках Allure Report указать директорию, в котором хранятся данные, сгенерированные адаптером для используемого тестового фреймворка, например, [allure-pytest](#)
5. После завершения сборки - посмотреть Allure-отчет в разделе меню Allure Report

The image features a background of a dense city skyline, likely New York City, viewed from an elevated perspective. The entire image is tinted with a blue and teal color palette. A network of thin, light blue lines connects various points across the image, creating a web-like pattern that overlays the cityscape. In the center, the word "Рефлексия" is written in a large, white, sans-serif font.


# Рефлексия

# Цели вебинара | После занятия вы сможете

1 Установить Allure локально и настроить отчетность в тестах

2 Настроить генерацию Allure-отчетов в Jenkins





Заполните, пожалуйста,  
опрос о занятии по ссылке в чате



O T U S

ОНЛАЙН ОБРАЗОВАНИЕ

Allure



Андрей Гридяев

Ведущий инженер по автоматизации тестирования

[andrey.gridyaev@gmail.com](mailto:andrey.gridyaev@gmail.com)