Pruebas en aplicaciones web y móvil con Selenium, JUnit 5, y Docker

Northem Quality
16/10/2019

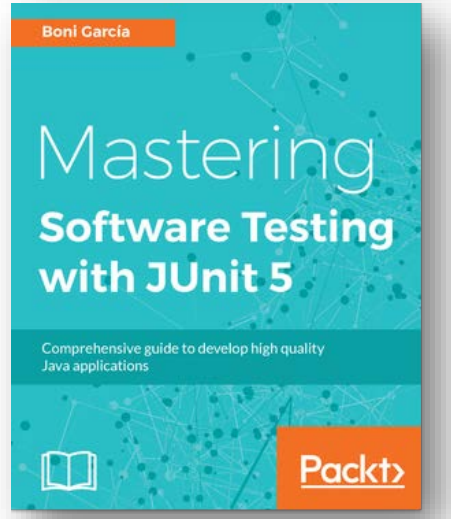
Boni García

 boni.garcia@urjc.es  http://bonigarcia.github.io/

 @boni_gg  https://github.com/bonigarcia

Boni García

- Assistant Professor at King Juan Carlos University (URJC) in Spain
- Author of 35+ research papers in different journals, magazines, international conferences, and the book Mastering Software Testing with JUnit 5
- Maintainer of different open source projects, such as WebDriverManager, Selenium-Jupiter, or DualSub

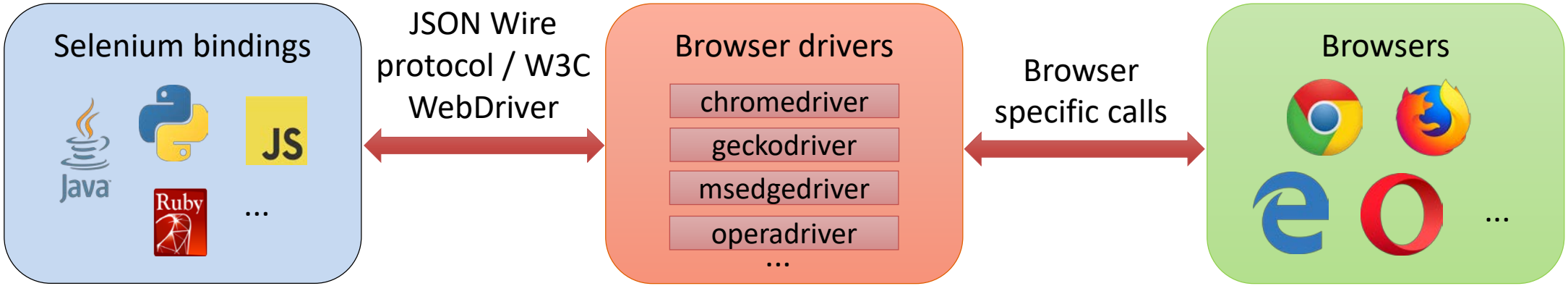


http://bonigarcia.github.io/

Table of Contents

1. Background - Selenium

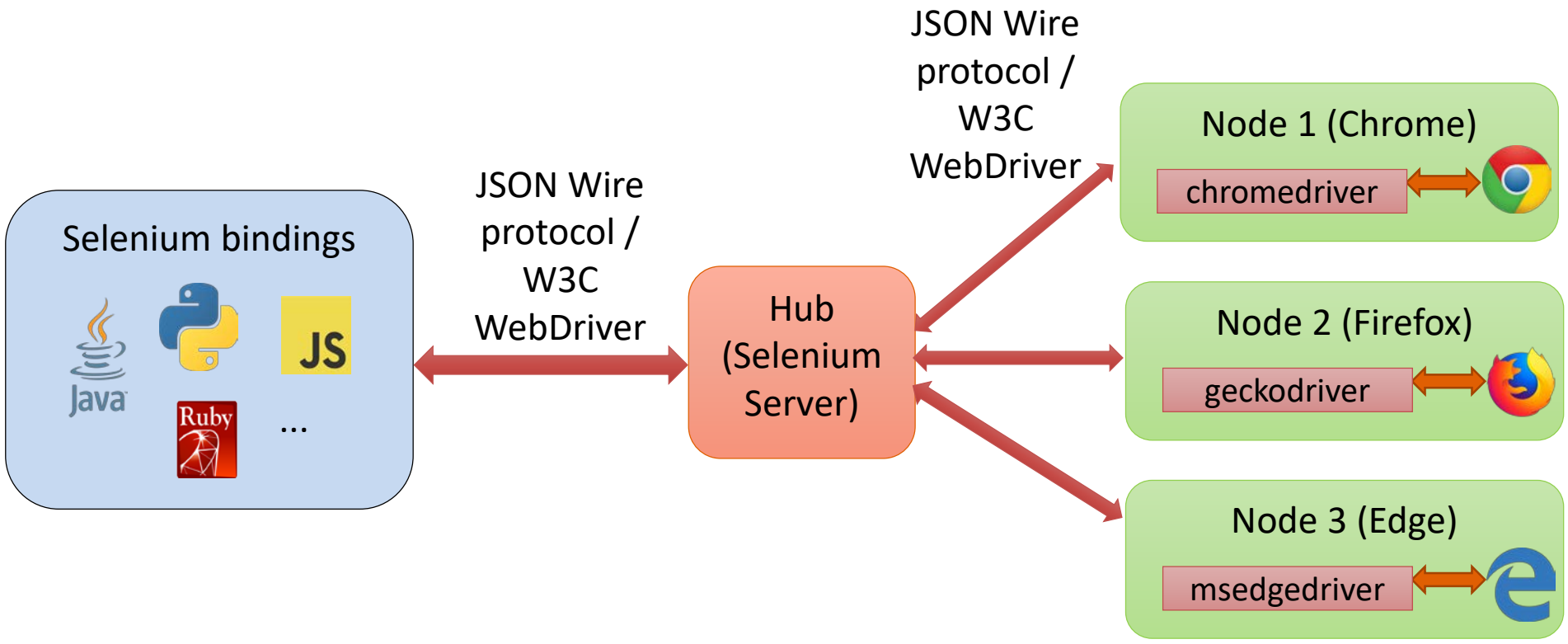
- **Selenium** is a family of projects for automated testing with browsers
  - **Selenium WebDriver** allows to control web browsers programmatically



https://seleniumhq.github.io/docs/site/en/webdriver/

1. Background - Selenium

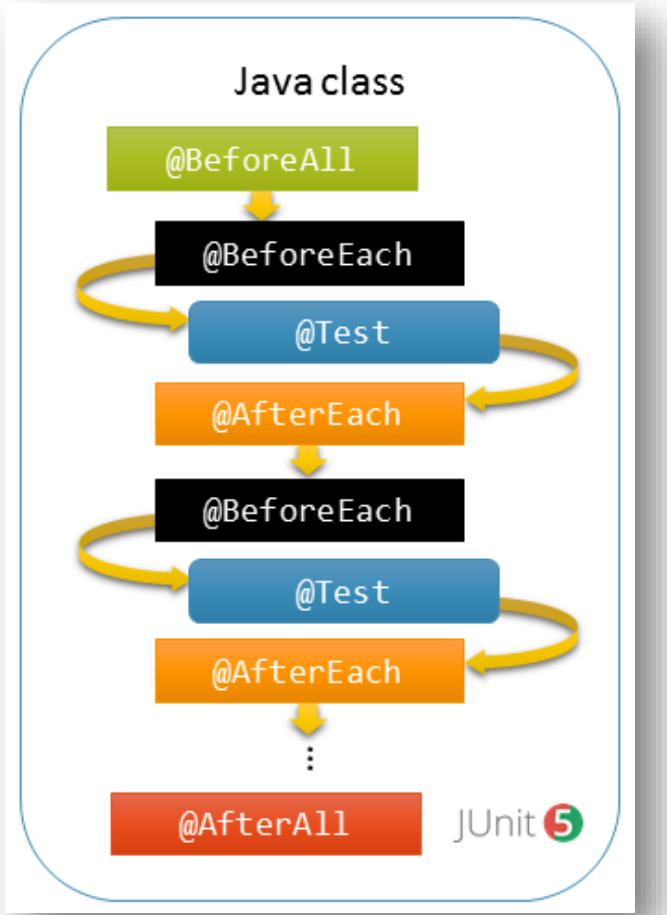
- **Selenium Grid** allows to drive web browsers in parallel hosted on remote machines:



https://seleniumhq.github.io/docs/site/en/grid/

1. Background - JUnit

- **JUnit** is the most popular testing framework for Java and can be used to implement different types of tests (unit, integration, end-to-end, …)
- **JUnit 5** (first GA released on September 2017) provides a brand-new programming an extension model called **Jupiter**

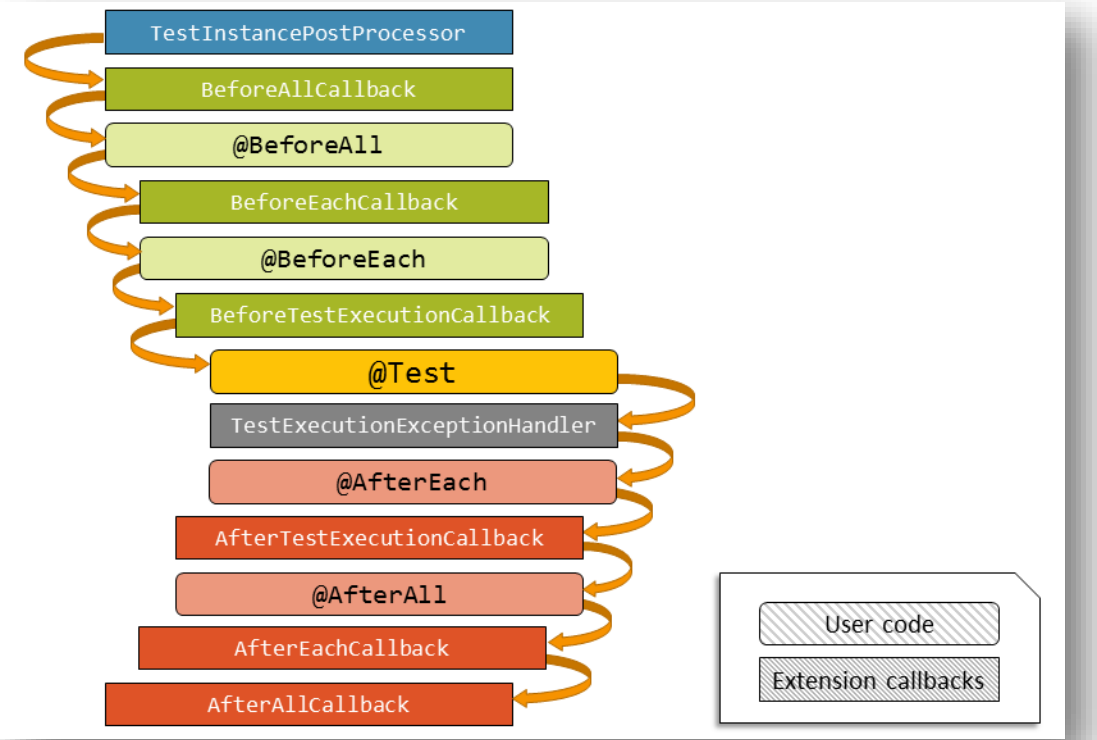


JUnit 5

https://junit.org/junit5/docs/current/user-guide/

1. Background - JUnit

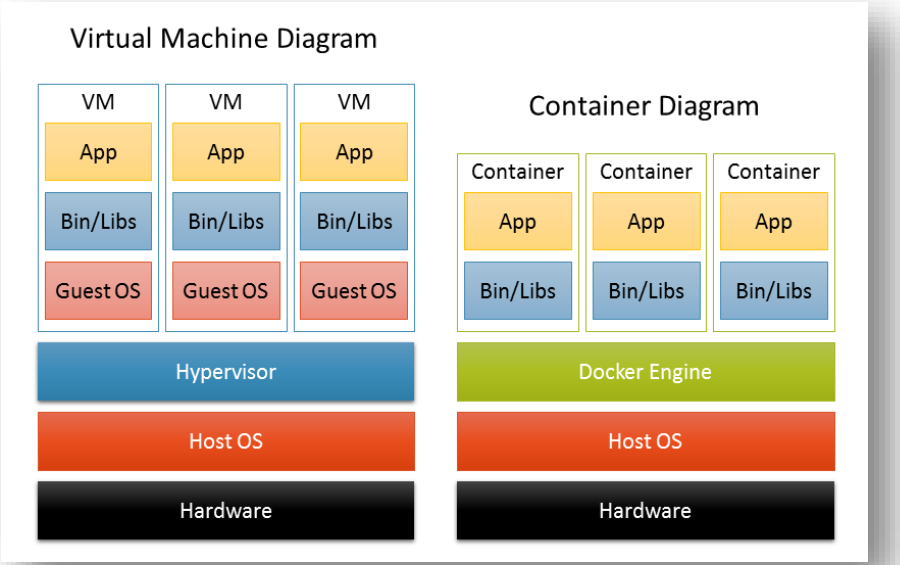
- The **extension model** of Jupiter allows to add custom features to the programming model:
  - Dependency injection in test methods and constructors
  - Custom logic in the test lifecycle
  - Test templates



Very convenient for Selenium!

1. Background - Docker

- **Docker** is a software technology which allows to pack and run any application as a lightweight and portable container
- The Docker platform has two main components: the Docker Engine, to create and execute containers; and the Docker Hub (https://hub.docker.com/), a cloud service for distributing containers



https://www.docker.com/

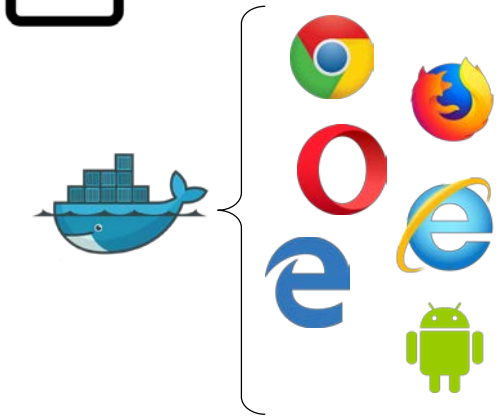
Table of Contents

2. Selenium-Jupiter - Motivation

- **Selenium-Jupiter** is a JUnit 5 extension aimed to ease the use of Selenium and Appium from Java tests

Clean test code (reduced boilerplate)

Effortless **Docker** integration (web browsers and Android devices)



Advanced features for tests

https://bonigarcia.github.io/selenium-jupiter/

2. Selenium-Jupiter - Setup

- **Selenium-Jupiter** can be included in a Java project as follows:

```xml
<dependency>
    <groupId>io.github.bonigarcia</groupId>
    <artifactId>selenium-jupiter</artifactId>
    <version>3.3.1</version>
    <scope>test</scope>
</dependency>
```

Maven

Using the latest version is always recommended!

```
dependencies {
    testCompile("io.github.bonigarcia:selenium-jupiter:3.3.1")
}
```

2. Selenium-Jupiter - Setup

- Source code: https://github.com/bonigarcia/selenium-jupiter
- Documentation: https://bonigarcia.github.io/selenium-jupiter
- Examples: https://github.com/bonigarcia/selenium-jupiter-examples

Requirements to run these examples:

- Java
- Maven/Gradle (alternatively some IDE)
- Docker Engine
- Linux (only required when running Android in Docker)

2. Selenium-Jupiter - Local browsers

- JUnit 4 and Selenium  JUnit 5 and Selenium-Jupiter:

2. Selenium-Jupiter - Local browsers

- Selenium-Jupiter uses JUnit 5's **dependency injection**:

Valid types: ChromeDriver, FirefoxDriver, OperaDriver, SafariDriver, EdgeDriver, InternetExplorerDriver, HtmlUnitDriver, PhantomJSDriver, AppiumDriver, SelenideDriver

```
@ExtendWith(SeleniumExtension.class)
class SeleniumJupiterTest {

    @Test
    void test(ChromeDriver chromeDriver) {
        // Use Chrome in this test
    }

}
```

2. Selenium-Jupiter - Local browsers

- Seamless integration with **Selenide** (fluent API for Selenium in Java):

https://selenide.org/

```
@ExtendWith(SeleniumExtension.class)
class SelenideDefaultTest {

    @Test
    void testWithSelenideAndChrome(SelenideDriver driver) {
        driver.open(
            "https://bonigarcia.github.io/selenium-jupiter/");
        SelenideElement about = driver.$(linkText("About"));
        about.shouldBe(visible);
        about.click();
    }

}
```

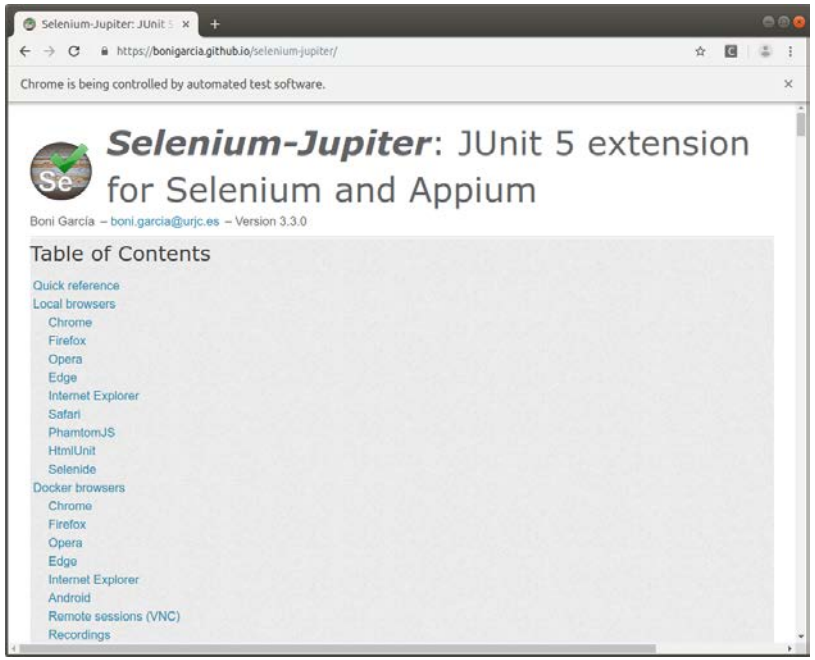
2. Selenium-Jupiter - Local browsers

- Use case: **WebRTC** applications (real-time communications using web browsers)
  - We need to specify **options** for browsers:

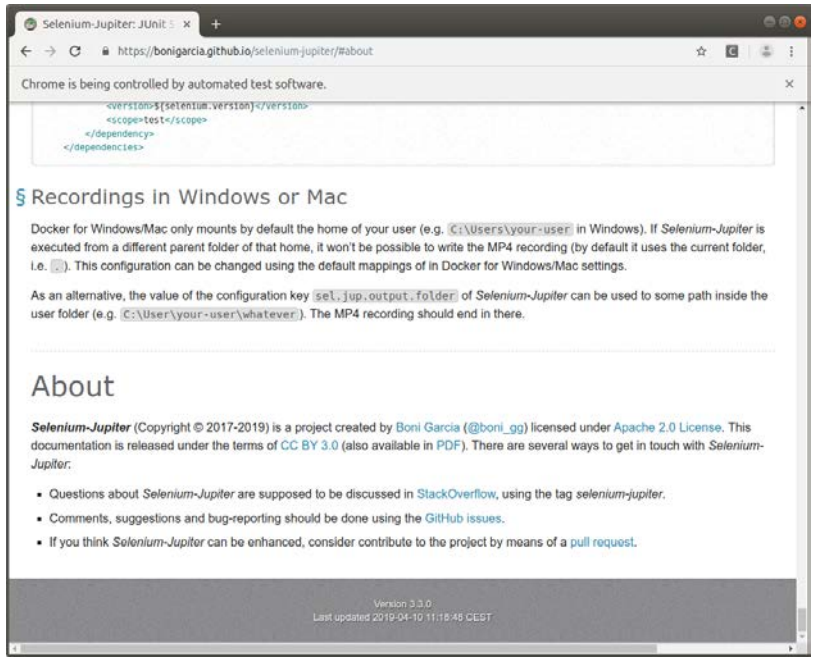
2. Selenium-Jupiter - Local browsers

- Use case: reuse same browser by different tests
  - Convenient for ordered tests (JUnit 5 new feature)

1

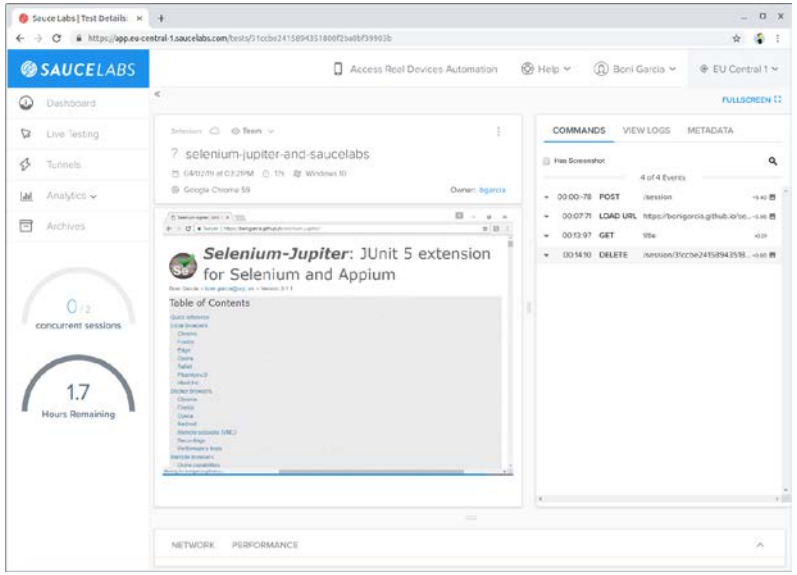


2



2. Selenium-Jupiter - Local browsers

- Selenium-Jupiter provides the annotations `@DriverUrl` and `@DriverCapabilities` to control remote browsers and mobiles, e.g.:



SAUCELABS
https://saucelabs.com/

appium
http://appium.io/

2. Selenium-Jupiter - Docker browsers

- Selenium-Jupiter provides seamless integration with **Docker** using the annotation `@DockerBrowser`:
  - Chrome, Firefox, and Opera:
    - Docker images for stable versions are maintained by Aerokube
    - Beta and unstable (Chrome and Firefox) are maintained by ElasTest
  - Edge and Internet Explorer:
    - Due to license, these Docker images are not hosted in Docker Hub
    - It can be built following a tutorial provided by Aerokube
  - Android devices:
    - Docker images for Android (docker-android project) by Budi Utomo

2. Selenium-Jupiter - Docker browsers

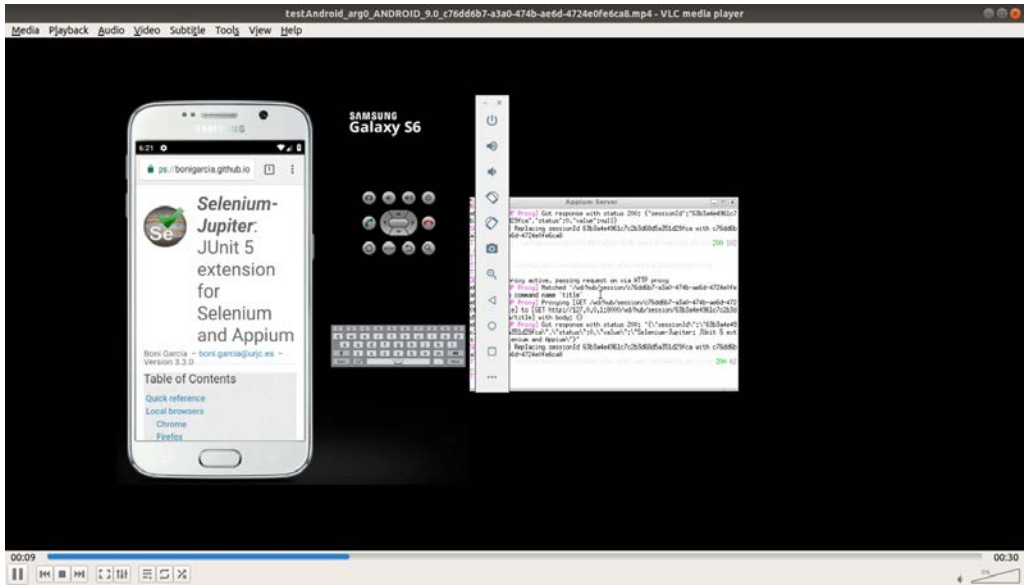
```
@ExtendWith(SeleniumExtension.class)
class DockerBasicTest {

    @Test
    void testFirefoxBeta(
            @DockerBrowser(type = FIREFOX, version = "beta") RemoteWebDriver driver) {
        driver.get("https://bonigarcia.github.io/selenium-jupiter/");
        assertThat(driver.getTitle(),
                containsString("JUnit 5 extension for Selenium"));
    }

}
```

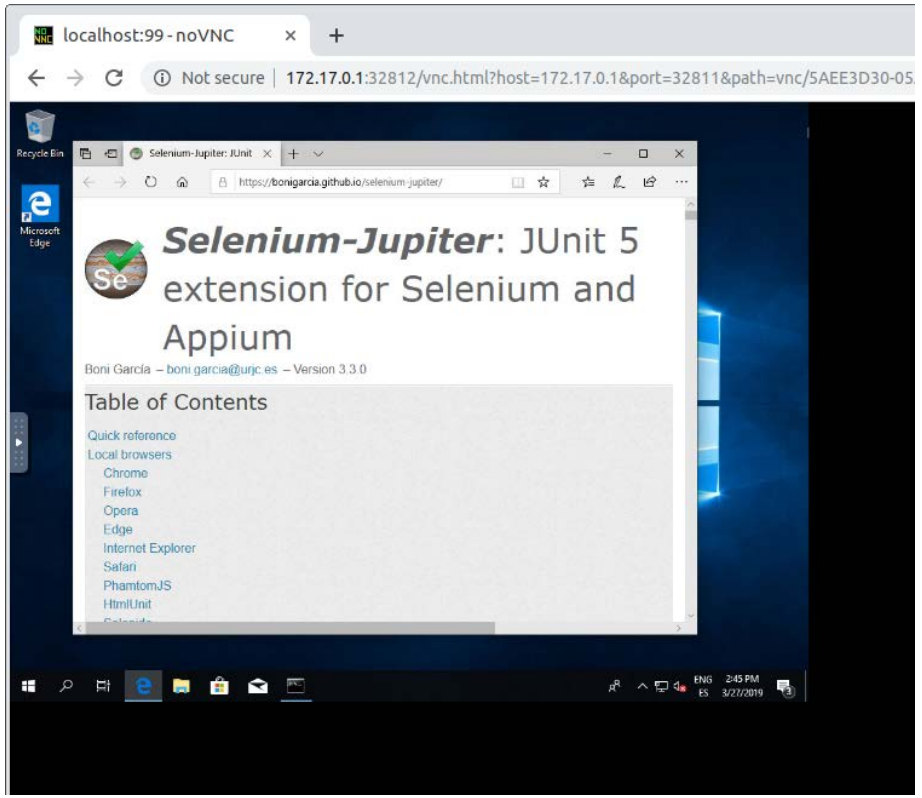
Supported browser types are: *CHROME*, *FIREFOX*, *OPERA*, *EDGE* , *IEXPLORER* and *ANDROID*

If *version* is not specified, the latest container version in Docker Hub is pulled. This parameter allows fixed versions and also the special values: *latest*, *latest-**, *beta*, and *unstable*

2. Selenium-Jupiter - Docker browsers

- The use of Docker enables a rich number of features:
  - Remote session access with **VNC**
  - Session **recordings**
  - **Performance** tests





2. Selenium-Jupiter - Docker browsers

- The possible **Android** setup options are the following:

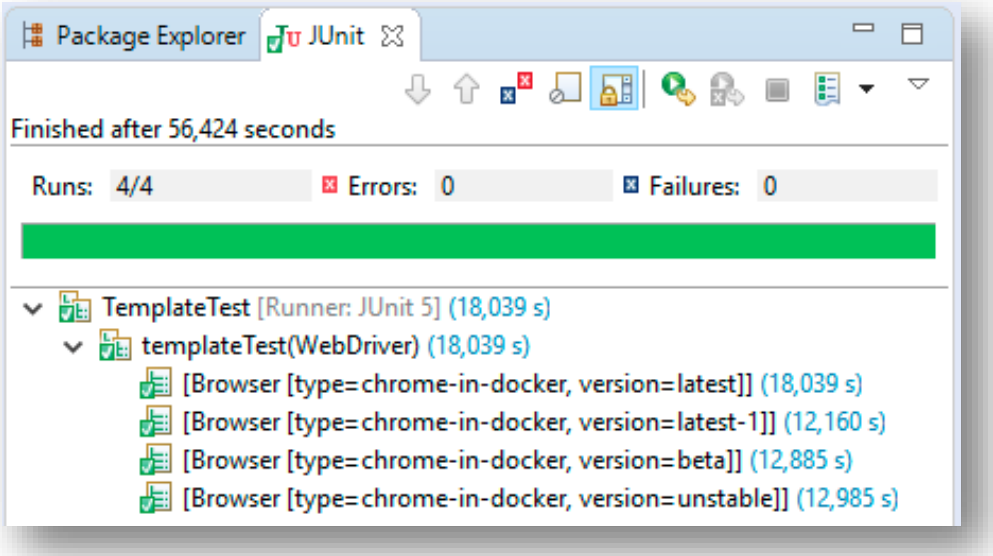
Type	Device name
Phone	Samsung Galaxy S6
Phone	Nexus 4
Phone	Nexus 5
Phone	Nexus One
Phone	Nexus S
Tablet	Nexus 7

Android version	API level	Browser name
5.0.1	21	browser
5.1.1	22	browser
6.0	23	chrome
7.0	24	chrome
7.1.1	25	chrome
8.0	26	chrome
8.1	27	chrome
9.0	28	chrome

2. Selenium-Jupiter - Test templates

- Selenium-Jupiter use the JUnit 5's support for **test templates**:

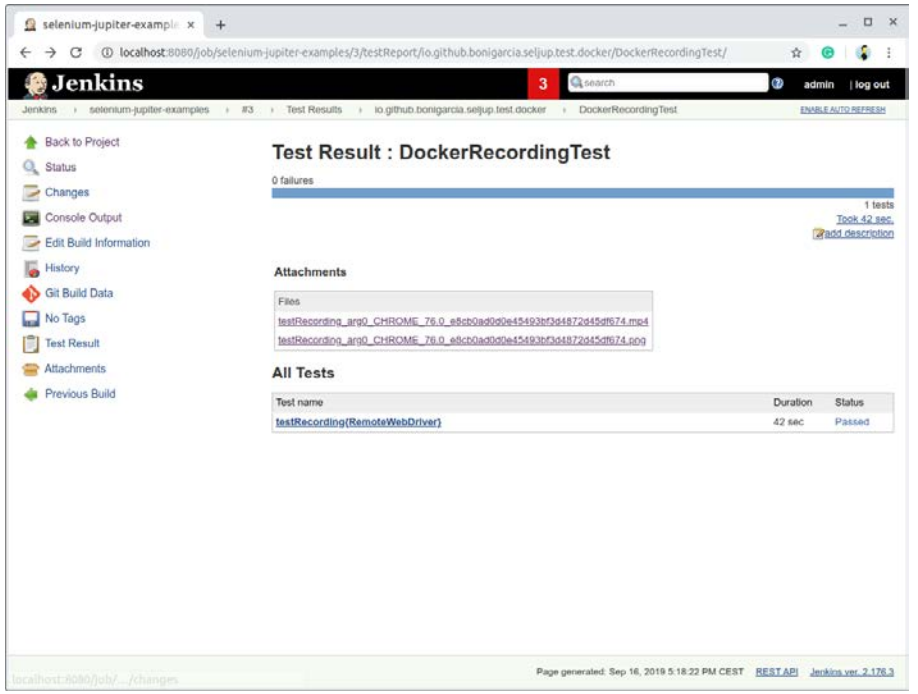
```
@ExtendWith(SeleniumExtension.class)
public class TemplateTest {

    @TestTemplate
    void templateTest(WebDriver driver) {
       // test
    }

}
```



2. Selenium-Jupiter - Integration with Jenkins

- Seamless integration with Jenkins through the Jenkins attachment plugin
- It allows to attach output files in tests (e.g. PNG screenshots and MP4 recordings) in the Jenkins GUI
- For example:

```
$ mvn clean test -Dtest=DockerRecordingTest \
    -Dsel.jup.recording=true \
    -Dsel.jup.screenshot.at.the.end.of.tests=true \
    -Dsel.jup.screenshot.format=png \
    -Dsel.jup.output.folder=surefire-reports
```



2. Selenium-Jupiter - Beyond Java

- Selenium-Jupiter can be also used:

1. As **CLI** (Command Line Interface) tool:

Selenium-Jupiter allows to control Docker browsers through VNC (manual testing)

```
$ java -jar selenium-jupiter-3.3.1-fat.jar chrome unstable
[INFO] Using Selenium-Jupiter to execute chrome unstable in Docker
...
```

2. As a **server** (using a REST-like API):

Selenium-Jupiter becomes into a Selenium Server (Hub)

```
$ java -jar webdrivermanager-3.3.1-fat.jar server
[INFO] Selenium-Jupiter server listening on http://localhost:4042/wd/hub
```

Table of Contents

3. Final remarks and future work

- Selenium-Jupiter has another features such as:
  - Configurable screenshots at the end of test (as PNG image or Base64)
  - Integration with Genymotion (cloud provider for Android devices)
  - Generic driver (configurable type of browser)
  - Mapping volumes in Docker containers
  - Access to Docker client to manage custom containers
- Selenium-Jupiter is in constant development. Its roadmap includes:
  - Implement a browser console (JavaScript log) gathering mechanism
  - Improve test template support (e.g. specifying options)
  - Improve scalability for performance tests (e.g. using Kubernetes)

Pruebas en aplicaciones web y móvil con Selenium, JUnit 5, y Docker

¡Muchas gracias!

Boni García

 boni.garcia@urjc.es  http://bonigarcia.github.io/

 @boni_gg  https://github.com/bonigarcia