



Devonfw Testing Guide

2018-01-29

Copyright © 2015-2018 the Devonfw Team, Capgemini

Table of Contents

1. Basics	1
1.1. Home	1
1.1.1. What is E2E Allure Test Framework	1
1.1.2. Benefits to the project	2
1.1.3. Road map plan	3
1.1.4. Wiki Structure:	3
1.2. How to install	4
1.2.1. Install can be done in two scenarios:	4
1.2.2. Modules	13
1.3. Core Test Module	13
1.3.1. What is Core Test Module	13
1.3.2. Core Test Module Functions	13
1.4. Selenium Test Module	14
1.4.1. What is Allure E2E Selenium Test Module	14
1.4.2. Selenium Structure	14
1.4.3. Framework Features	14
1.5. WebAPI Test Module	16
1.6. Security Test Module	17
1.6.1. Scope definition	17
1.7. Mobile Test Module	19
1.8. Standalone Test Module	20
1.9. DevOps Module	21
1.9.1. How we see DevOps	21
1.9.2. QA Team Goal	21
1.9.3. Well rounded test case production process	21
1.9.4. What will you receive in this DevOps module	22
1.9.5. What will you gain with our DevOps module	22
1.9.6. How to build this DevOps module	25

1. Basics

1.1 Home

1.1.1 What is E2E Allure Test Framework

End to end automation test framework written in Java.

E2E Test Framework for DevOps & Smart Automation

Task Description

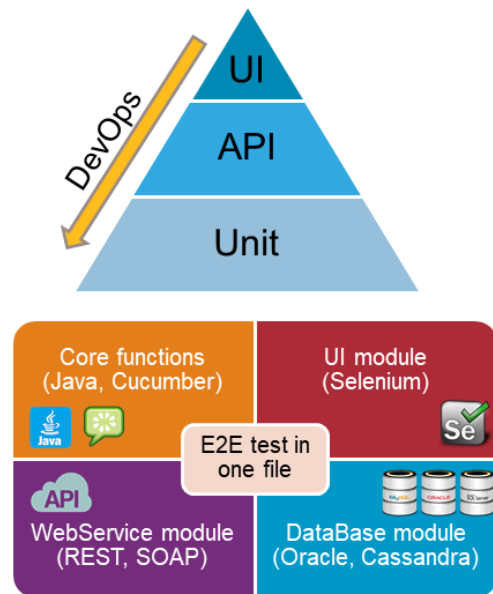
- Embrace Test Automation in project with minimal cost and time
- Bring up to speed testers and developers with common quality goal
- Plug in/out framework features to fit project needs
- One test case to validate 3 tiers of app (Database, Webservice, UI)
- Accelerate tests execution (parallel execution, env auto-scaling)
- Standardize test develop process

Service provided by Capgemini

- Takeover of operation and therefore delivery of the entire service in accordance with DevOps procedural model.
- Capgemini is partnering with client on implementation of IT Projects Delivery Transformation
- Tool agnostic approach / Innovative architecture – Test Framework with examples
- Test execution infrastructure (Jenkins as code pipeline, Docker images)
- Scalable and resilient UI web browser environment (Selenium Grid)

Benefit for the Customer

- Resilient and robust building and validation process
- Shift quality gates closer to software developer process
- Team quality awareness increase- include Unit Tests, Static Analyze, Security Tests, Performance
- Transparent test execution environment to any infrastructure. Touch base with Cloud solution
- Faster Quality and DevOps driven delivery
- Proven frameworks, technologies and processes



E2E Test Framework modules



* - task not started, to be defined

1.1.2 Benefits to the project

Benefits to project

End to end project verification under one solution

Resilient and robust DevOps driven delivery

Team quality awareness

Shift left quality gates

Easy to enhance

1.1.3 Road map plan

Allure E2E Framework road map



1.1.4 Wiki Structure:

- [How to install Allure Test Framework](#)
- Allure Test Framework modules:
 - [Core test module](#)
 - [Selenium test module](#)
 - [WebAPI test module](#)
 - [Security test module](#)
 - [DataBase test module](#)
 - [Mobile test module](#)
 - [Standalone test module](#)
 - [DevOps module](#)

1.2 How to install

1.2.1 Install can be done in two scenarios:

- Out of the box install - Fast and easy
- Advanced installation - How to install all ingredients

Easy out of the box install

1. Java 1.8 JDK 64bit

- Download and install [Java download link](#)

Java SE Development Kit 8u131		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
<input checked="" type="radio"/> Accept License Agreement <input type="radio"/> Decline License Agreement		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.87 MB	jdk-8u131-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.81 MB	jdk-8u131-linux-arm64-vfp-hflt.tar.gz
Linux x86	164.66 MB	jdk-8u131-linux-i586.rpm
Linux x86	179.39 MB	jdk-8u131-linux-i586.tar.gz
Linux x64	162.11 MB	jdk-8u131-linux-x64.rpm
Linux x64	176.95 MB	jdk-8u131-linux-x64.tar.gz
Mac OS X	226.57 MB	jdk-8u131-macosx-x64.dmg
Solaris SPARC 64-bit	139.79 MB	jdk-8u131-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.13 MB	jdk-8u131-solaris-sparcv9.tar.gz
Solaris x64	140.51 MB	jdk-8u131-solaris-x64.tar.Z
Solaris x64	96.96 MB	jdk-8u131-solaris-x64.tar.gz
Windows x86	191.22 MB	jdk-8u131-windows-i586.exe
Windows x64	198.03 MB	jdk-8u131-windows-x64.exe

- Windows Local Environment [how to set](#):
 - **Variable name:** JAVA_HOME | **Variable value:** c:\Where_You've_Installed_Java
 - **Variable name:** PATH | **Variable value:** %JAVA_HOME%\bin;%JAVA_HOME%\lib



- Verify in command line:

```
> java --version
```

2. Download package [Ready to use AllureTestEnvironment](#)
3. To folder C:\ unzip with [7z](#) downloaded Allure Test Framework

Note: Please double check, place where you have unzipped Allure_Test_Framework



4. In unzipped folder (C:\Allure_Test_Framework\workspace) run *start-eclipse.bat*
5. . Update project structure (*ALT + F5*)



Manual step by step install

1. Java 1.8 JDK 64bit
 - Download and install [Java download link](#)

Java SE Development Kit 8u131

You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

☒ **Accept License Agreement**
☐ **Decline License Agreement**

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.87 MB	jdk-8u131-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.81 MB	jdk-8u131-linux-arm64-vfp-hflt.tar.gz
Linux x86	164.66 MB	jdk-8u131-linux-i586.rpm
Linux x86	179.39 MB	jdk-8u131-linux-i586.tar.gz
Linux x64	162.11 MB	jdk-8u131-linux-x64.rpm
Linux x64	176.95 MB	jdk-8u131-linux-x64.tar.gz
Mac OS X	226.57 MB	jdk-8u131-macosx-x64.dmg
Solaris SPARC 64-bit	139.79 MB	jdk-8u131-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.13 MB	jdk-8u131-solaris-sparcv9.tar.gz
Solaris x64	140.51 MB	jdk-8u131-solaris-x64.tar.Z
Solaris x64	96.96 MB	jdk-8u131-solaris-x64.tar.gz
Windows x86	191.22 MB	jdk-8u131-windows-i586.exe
Windows x64	198.03 MB	jdk-8u131-windows-x64.exe

- Windows Local Environment [how to set](#):
 - Variable name:** JAVA_HOME | **Variable value:** c:\Where_You've_Installed_Java
 - Variable name:** PATH | **Variable value:** %JAVA_HOME%\bin;%JAVA_HOME%\lib

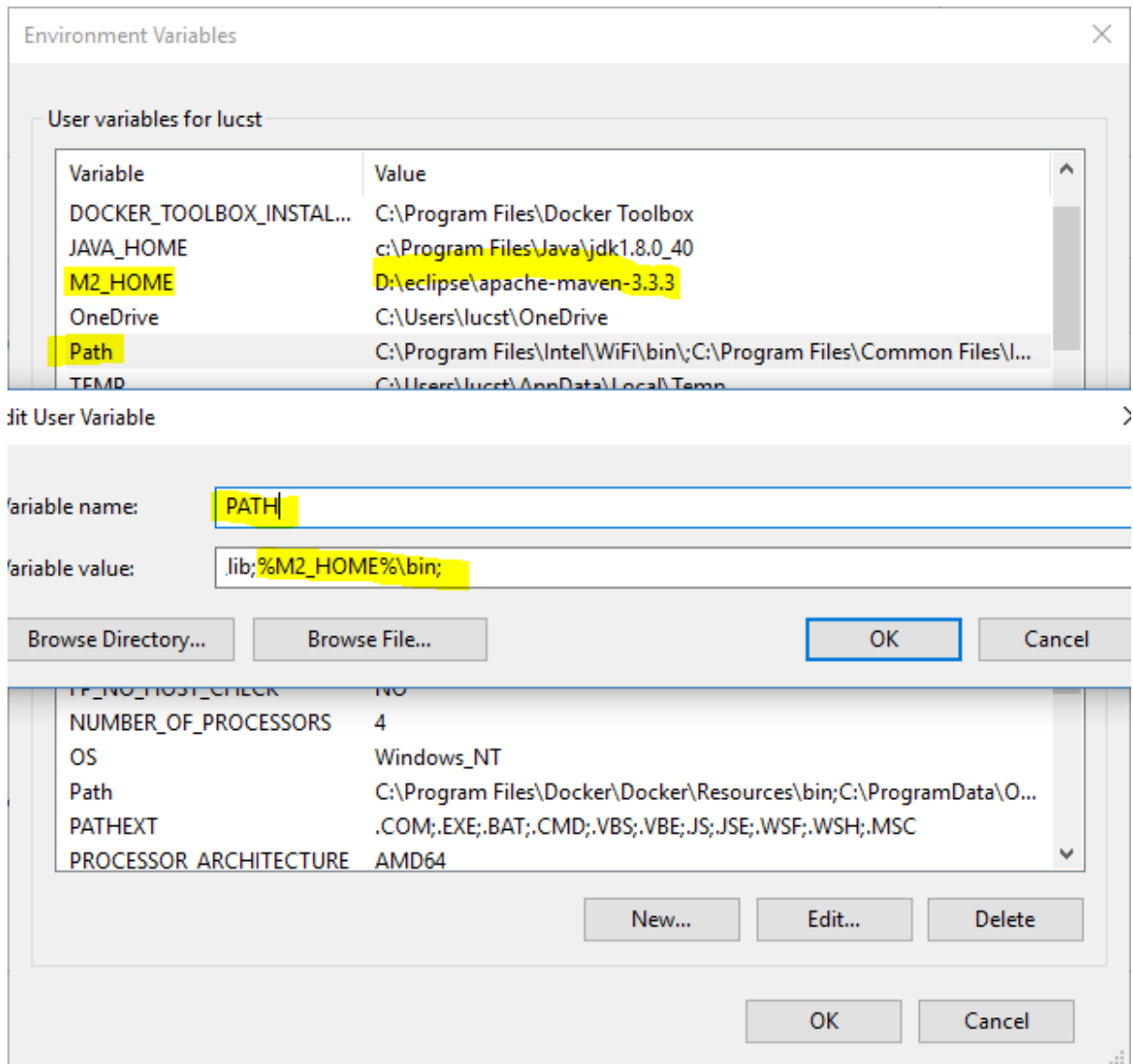


- Verify in command line:

```
> java --version
```

2. Maven 3.5

- Download Maven <http://www-eu.apache.org/dist/maven/maven-3/3.5.0/binaries/apache-maven-3.5.0-bin.zip>
- Unzip, to C:\maven
- Windows Local Environment
 - **Variable name:** M2_HOME | **Variable value:** c:\maven
 - **Variable name:** PATH | **Variable value:** %M2_HOME%\bin



- Verify in command line:

```
> mvn --version
```

3. Eclipse IDE

- Download and unzip [Eclipse](#)

4. Download Allure Test Framework [source code](#)

5. Import projects in Eclipse

- Import:



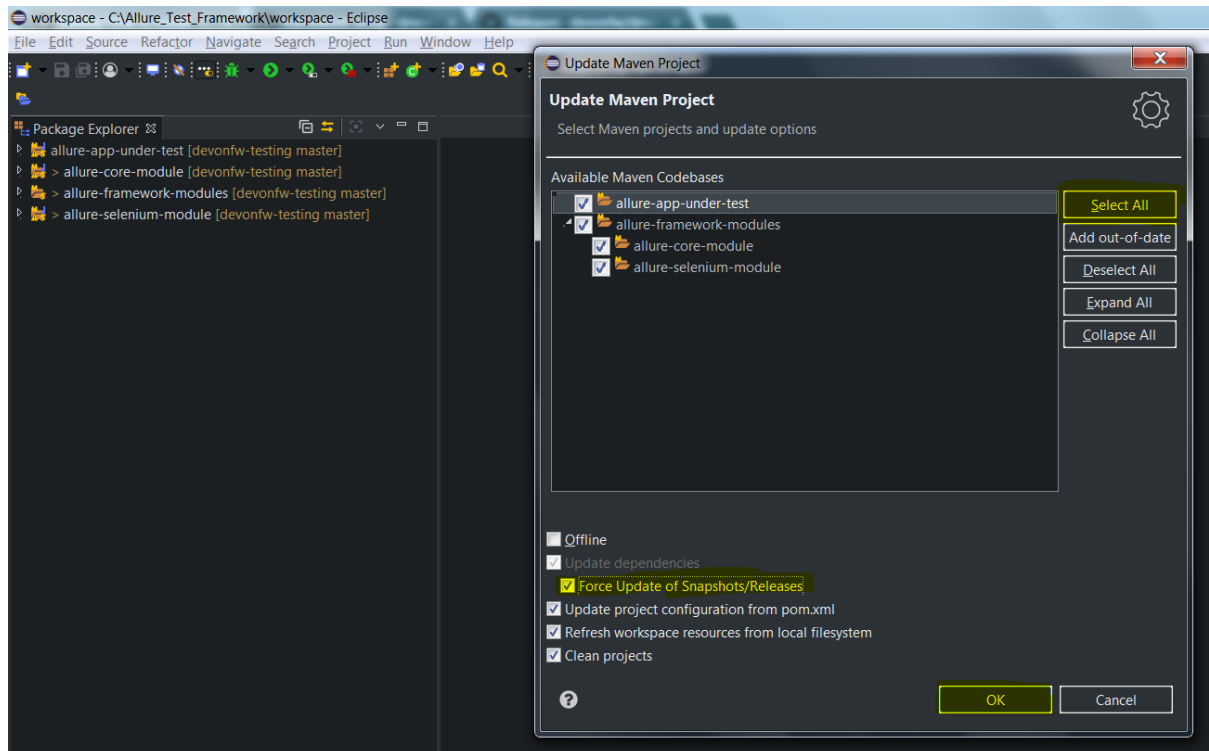
- Projects from folders:



- Open already created projects



- Update project structure - *ALT + F5*

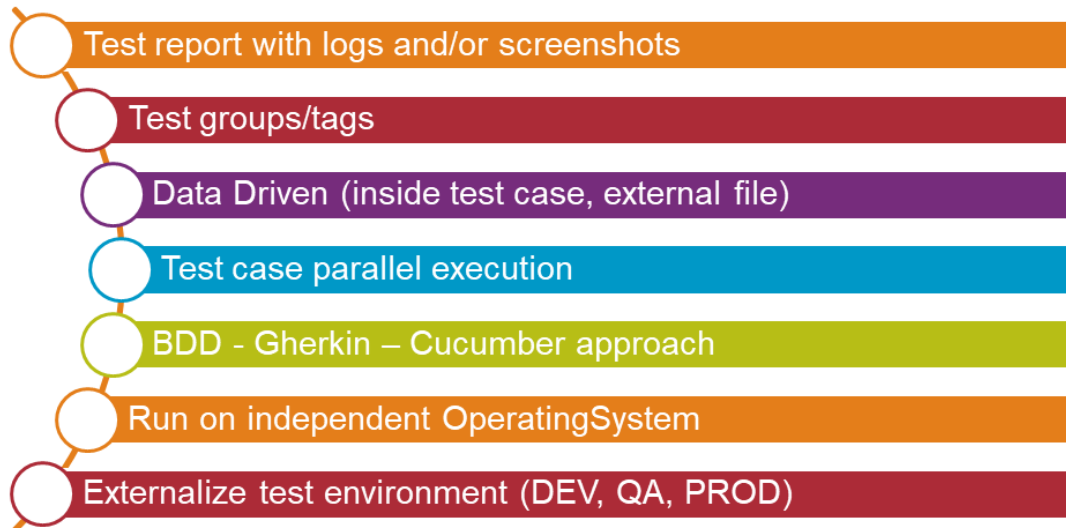


1.2.2 Modules

1.3 Core Test Module

1.3.1 What is Core Test Module

Core functionality ingredients



1.3.2 Core Test Module Functions

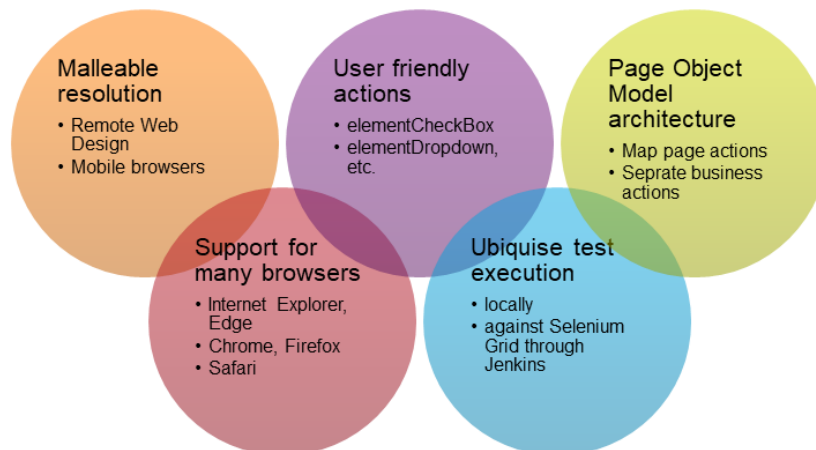
- [Test reports with logs and/or screenshots](#)
- [Test groups/tags](#)
- [Data driven approach](#)
- [Test case parallel execution](#)
- BDD - Gherkin - Cucumber approach
- [Run on independent Operating Systems](#)
- [Externalize test environment \(DEV, QA, SIT, PROD\)](#)

[[core-test-module_how-to-start?]] == How to start? Read: [Framework Test Class](#)

1.4 Selenium Test Module

1.4.1 What is Allure E2E Selenium Test Module

UI Selenium test module ingredients



1.4.2 Selenium Structure

- [What is Selenium](#)
- [What is WebDriver](#)
- [What is Page Object Model/Pattern](#)
- [List of web elements \(Button, Dropdown, Checkbox, Alert Popup, etc.\)](#)

1.4.3 Framework Features

- [Construction of Framework Page Class](#)
 - Every Page class must extend BasePage
 - What is isLoaded(), load() and pageTitle() for
 - How to create selector variable - 'private static final By ButtonOkSelector = By.Css(...)'
 - How to prepare everlasting selector - [documentation](#)
 - Method/action naming convention - [documentation](#)
 - Why we should use findDynamicElement() and findElementQuietly() instead of classic Selenium findElement

- List of well-rounded groups of user friendly actions (ElementButton, ElementCheckbox, ElementInput, etc.)
- Verification points of well-defined Page classes and Test classes - [documentation](#)
- [Run on different browsers: Chrome, Firefox, IE, Safari, Edge](#)
- [Run with full range of resolution \(mobile and desktop\): Testing Response Design Webpage](#)

[[selenium-test-module_how-to-start?]] == How to start? Read: [My first Selenium Test](#)

1.5 WebAPI Test Module

1.6 Security Test Module

[[security-test-module_what-is-security?]] == What is Security?

Application Security is concerned with **Integrity**, **Availability** and **Confidentiality** of data processed, stored and transferred by the application.

Application Security is a cross-cutting concern which touches every aspect of the Software Development Lifecycle. You can introduce some SQL injection flaws in your application and make it exploitable, but you can also expose your secrets due to poor secret management process (which will have nothing to do with code itself), and fail as well.

Because of this, and many other reasons, not every aspect of security can be automatically verified. Manual tests and audits will be still needed. Nevertheless, every security requirement which are automatically verified, will prevent code degeneration and misconfiguration in a continuous manner.

[[security-test-module_how-to-test-security?]] == How to test Security?

Security tests can be performed in many different ways like:

- **Static Code Analysis** - improves the security by (usually) automated code review. Good way to search after vulnerabilities, which are 'obvious' on the code level (like e.g. SQL injection). The downside is that the professional tools to perform such scans are very expensive and still produce many false positives.
- **Dynamic Code Analysis** - tests are run against a working environment. Good way to search after vulnerabilities, which require all client- and server-side components to be present and running (like e.g. Cross-Site Scripting). Tests are performed in a semi-automated manner and require a proxy tool (like e.g. OWASP ZAP)
- **Unit tests** - self written and maintained tests. They work usually on the HTTP/REST level (as this defines the trust boundary between the client and the server) and run against a working environment. Unit tests are best suited to verify requirements which involve business knowledge of the system or which assure secure configuration on the HTTP level.

In the current release of the Security Module the main focus will be **Unit Tests**.

Although the most common choice of environment for security tests to run on will be **integration** (as the environment offers the right stability and should mirror the production closely), it is not uncommon for some security tests to run on production as well. This is done for e.g. TLS configuration testing to ensure proper configuration of the most relevant environment in a continuous manner.

1.6.1 Scope definition

Unresolved directive in DevonfwTesting.asciidoc - include::Database-test-module.asciidoc[]

1.7 Mobile Test Module

1.8 Standalone Test Module

1.9 DevOps Module

1.9.1 How we see DevOps

DevOps consists of a mix of three key components in a technical project: * People skills and mind set
* Processes * Tools

In **E2E Allure Test Framework** we would like to address majority of these components.

1.9.2 QA Team Goal

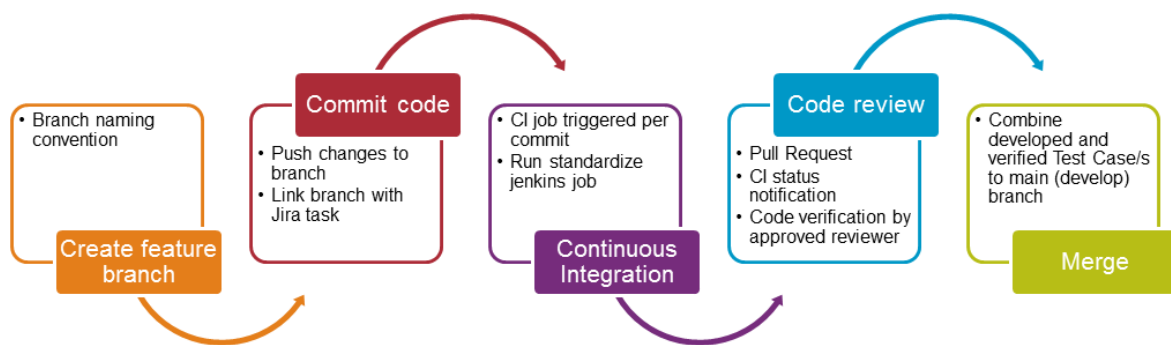
As QA engineers we always take a lot of care about product code quality.

Therefore, we also have to understand, **test case is also a code which has to be validated** against quality gates. As a result, we must **test our developed test case** just as it was done during standard Software Delivery Life Cycle.

1.9.3 Well rounded test case production process

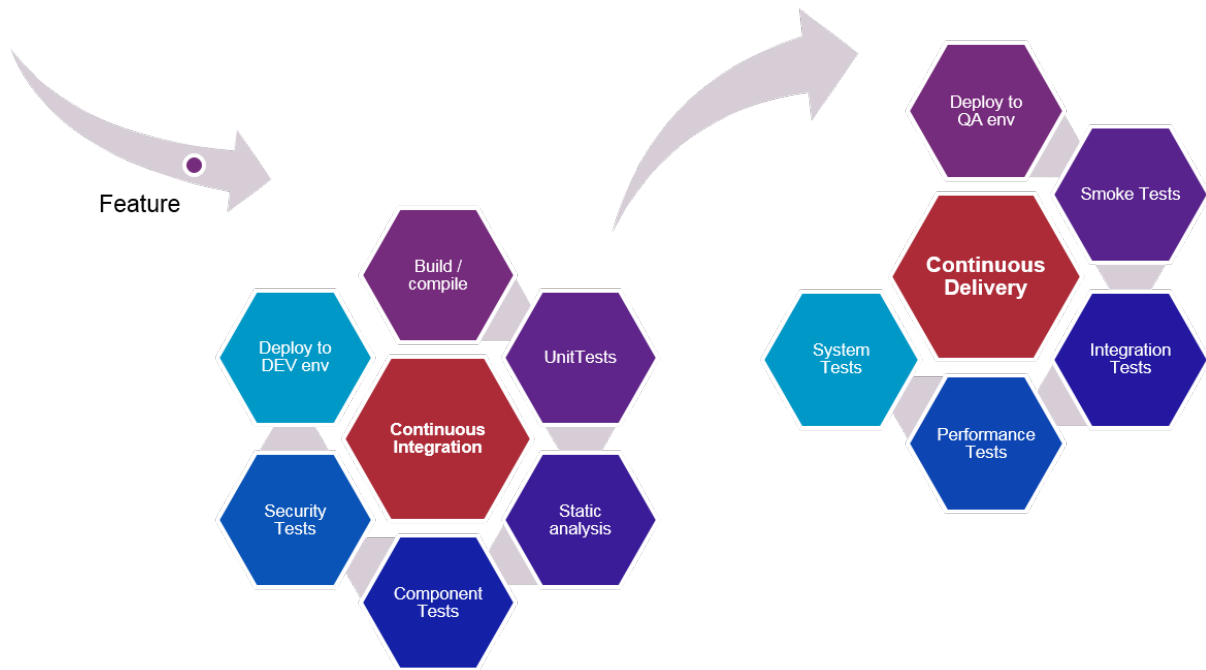
- How we define top notch test cases develop process in **E2E Allure Test Framework**

Well defined Test Case develop process



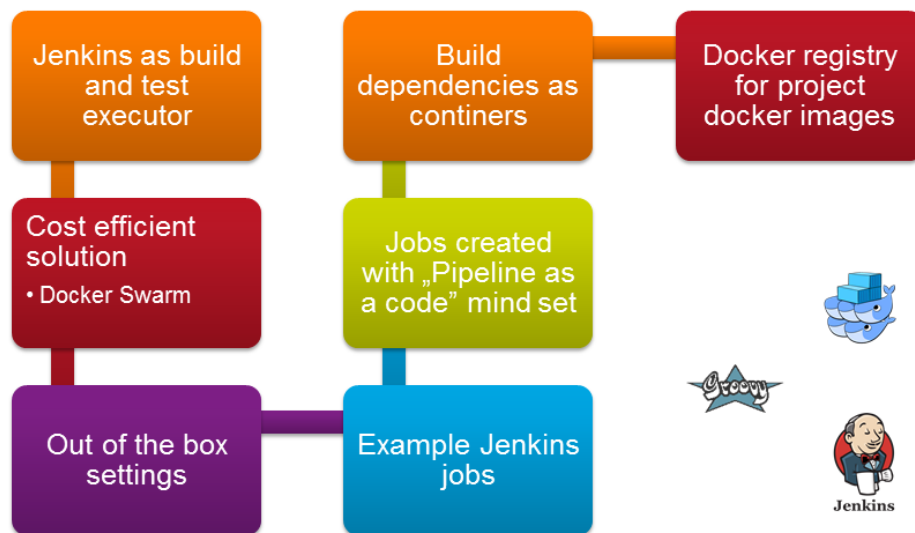
[[devops-module_continuous-integration-(ci)-and-continuous-delivery-(cd)]] == Continuous Integration (CI) and Continuous Delivery (CD)

- [Continuous Integration \(CI\)](#) - procedure where quality gates validate test case creation process
- [Continuous Delivery \(CD\)](#) - procedure where we include as smoke/regression/security created test cases, validated against CI



1.9.4 What will you receive in this DevOps module

DevOps infrastructure ingredients

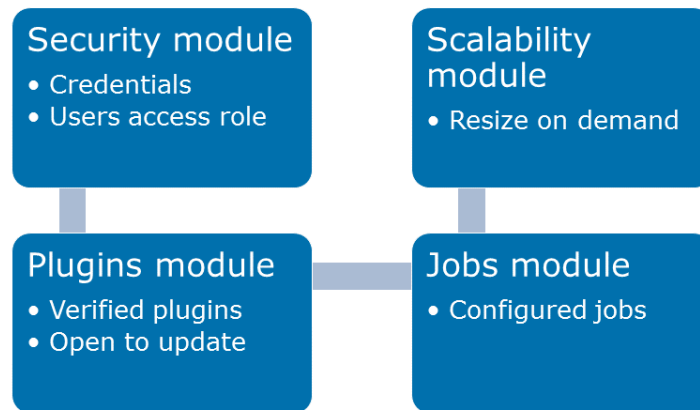


1.9.5 What will you gain with our DevOps module

The CI procedure has been divided into transparent modules. This solution makes configuration and maintenance very easy because you can manage versions and customize the configuration

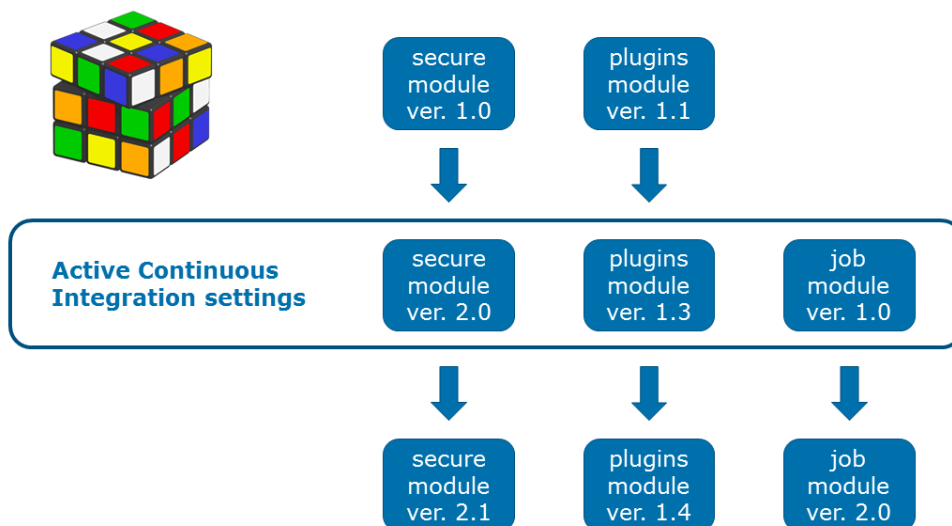
independently for each module. A separate security module ensures protection of your credentials and assigned access roles regardless of changes in other modules.

Superior Continuous Integration ingredients



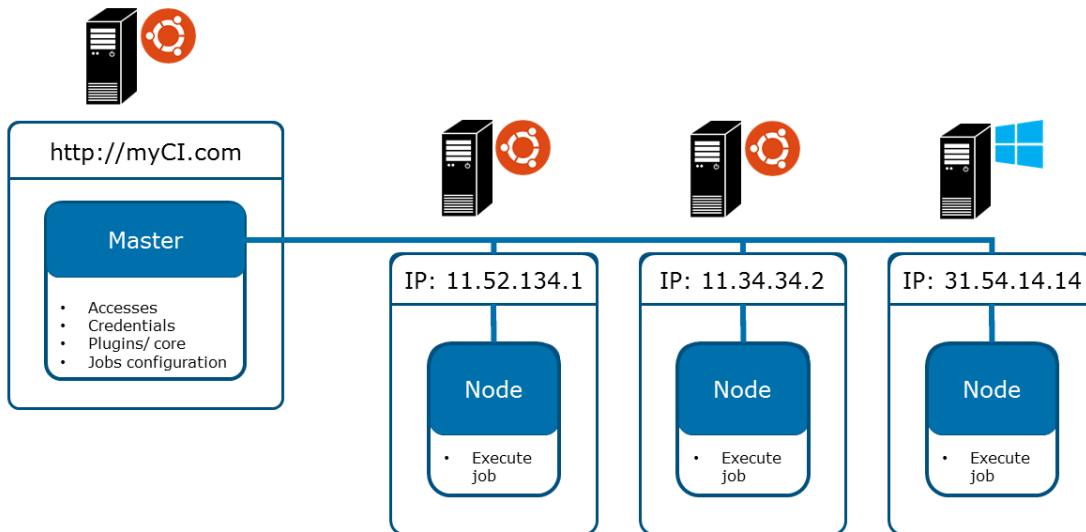
Your CI process will be matched to the current project. You can easily go back to the previous configuration, test a new one or move a selected one to other projects.

Setup your own proven CI modules

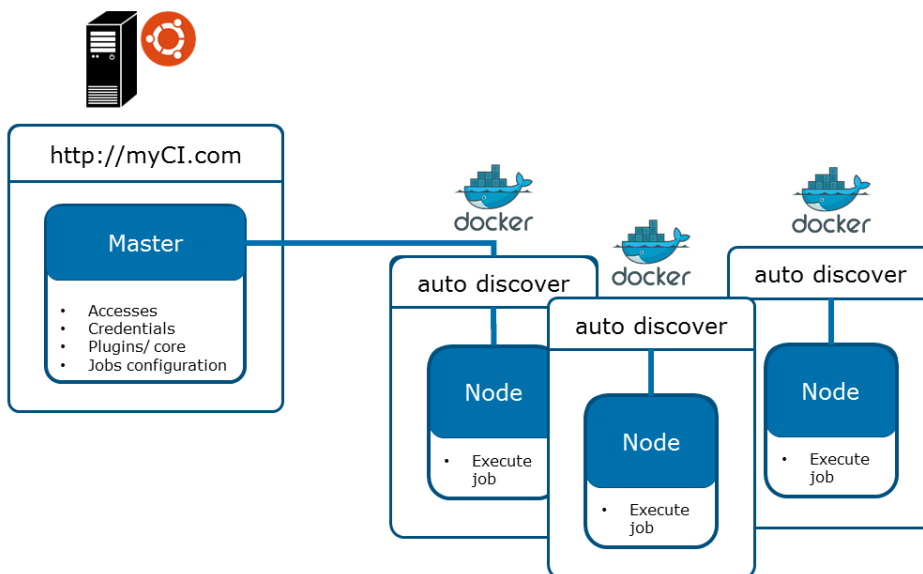


DevOps module supports delivery model in which executors are made available to the user as needed. It has advantages such as: * Save computing resources * Eliminate guessing on your infrastructure capacity needs * Stop spending time on running and maintaining additional executors

Classic executor architecture for Continuous Integration



Innovative executor architecture for Continuous Integration



Benefits



Modularity

Build your own CI



Secure

Restricted access to credentials



Autoscaling and service discovery

Take as you need

1.9.6 How to build this DevOps module

If you want to install the module, please click the link below. Installation should not take more than a few minutes * [DevOps module installation](#)

Once you have implemented the module, you can learn more about:

- [Building jobs & Running builds](#)
- [Docker commands](#)