Software Project Management - Laboratory

Lecture n° 11
A.Y. 2021-2022

Prof. Fabrizio Fornari

Apache Maven

Apache Maven is an open source, standards-based project management framework that simplifies the building, testing, reporting, and packaging of projects.

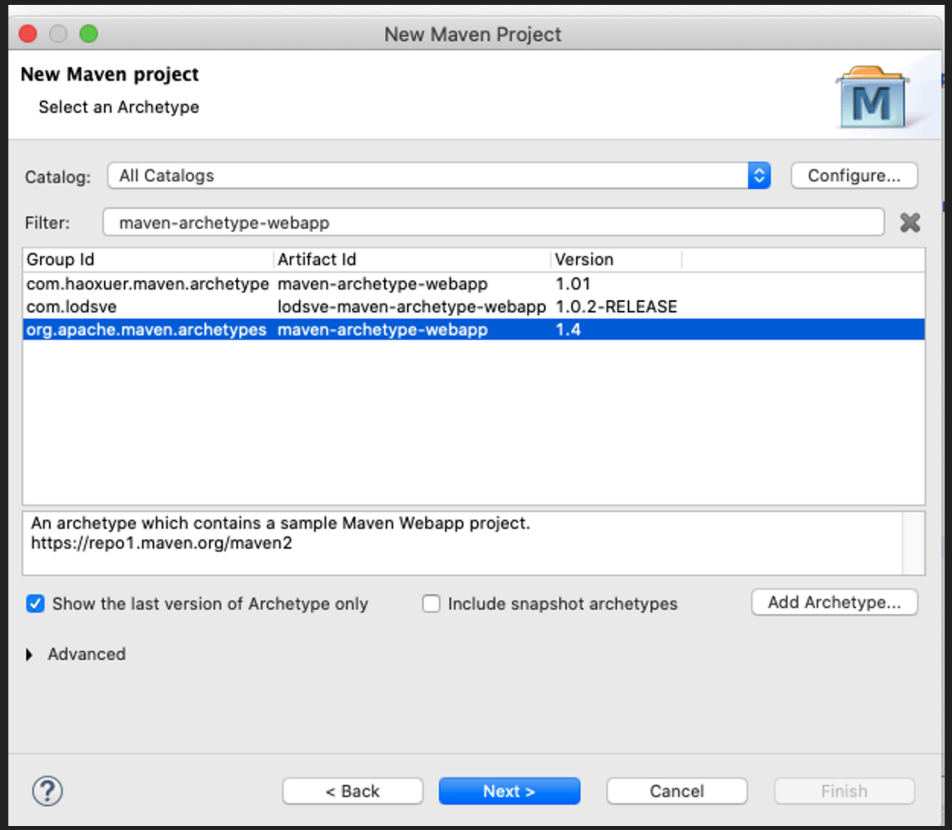
http://maven.apache.org/

Maven - Archetypes

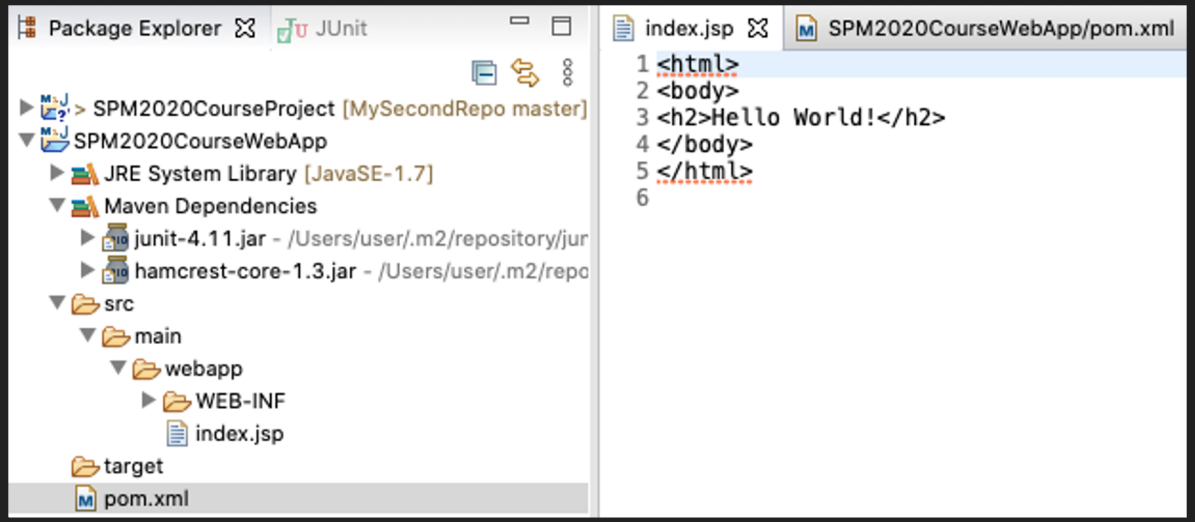
Maven archetypes are project templates that allow users to generate new projects easily

Create a Maven Project by following: File → New → Other → Maven Project → Next

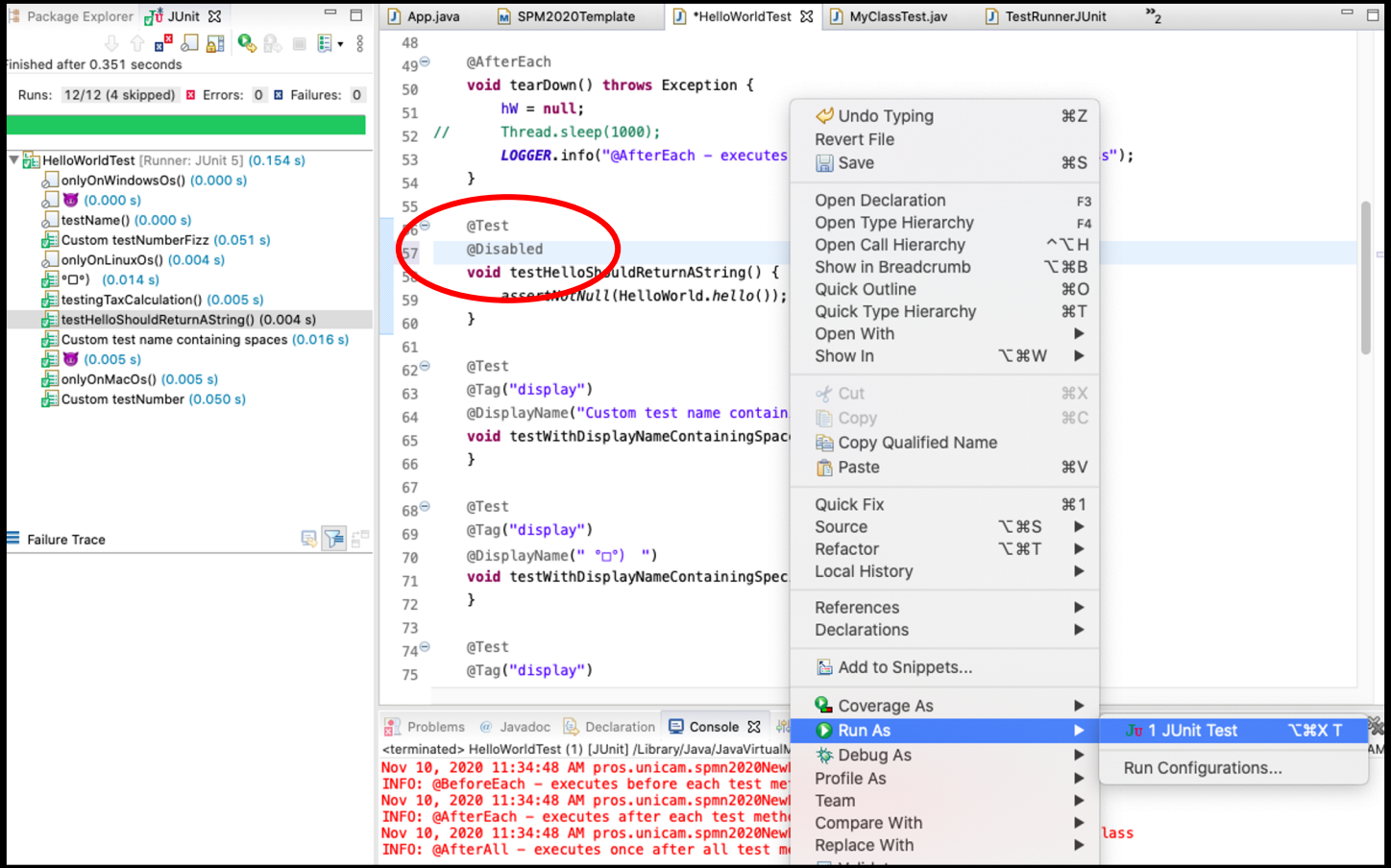
Insert “maven-archetype- webapp”, select and proceed



Maven - Archetype WebApp

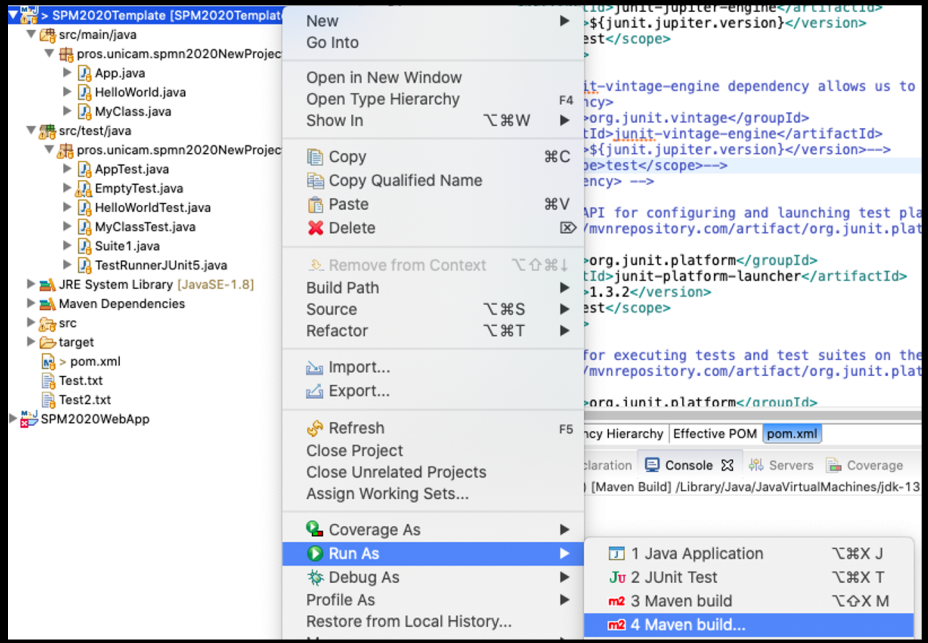


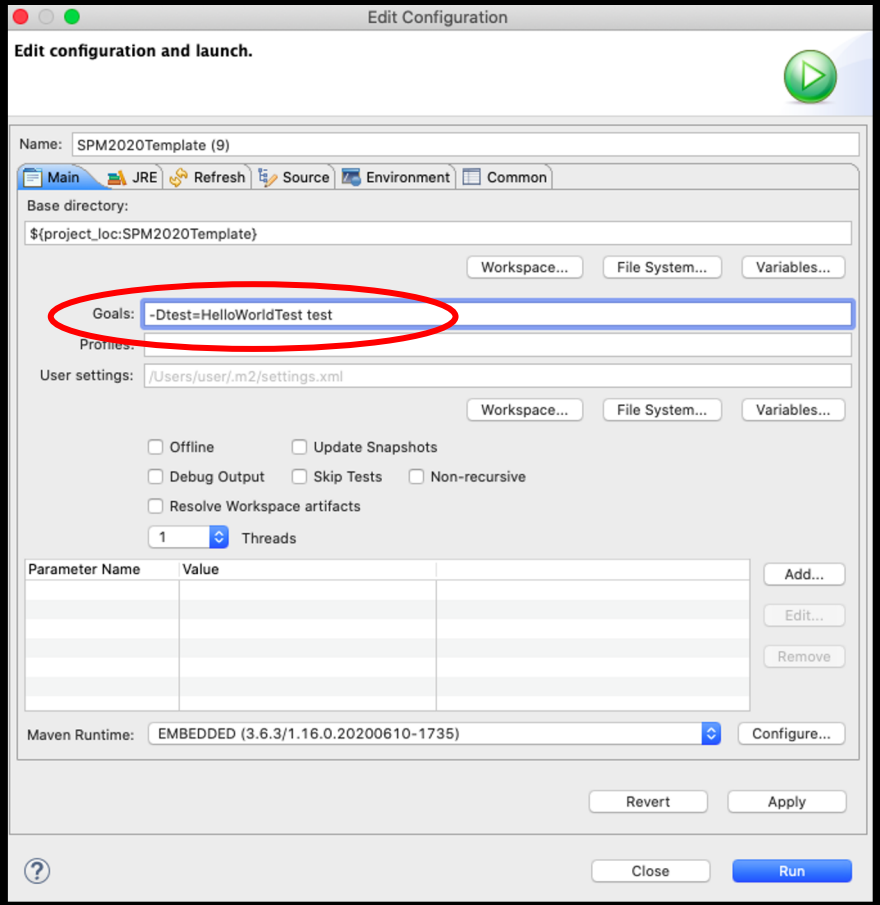
Manual Testing in Eclipse



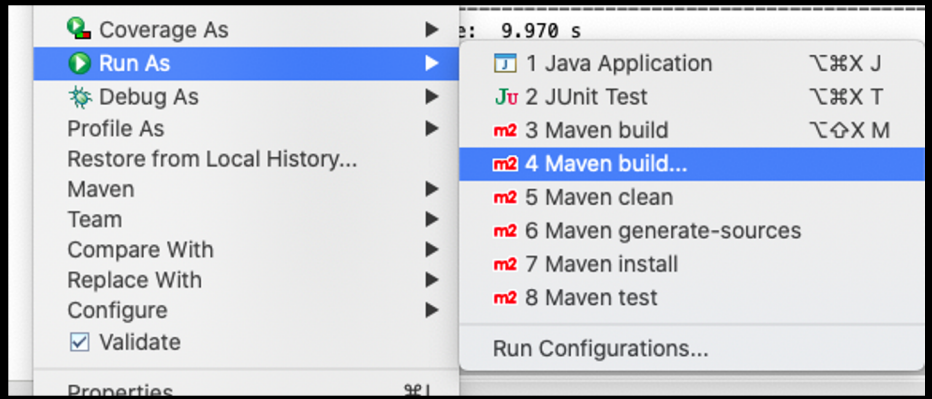
Manual Testing with Maven

- Run a single test class: -Dtest=<NameOfTheTestClass> test

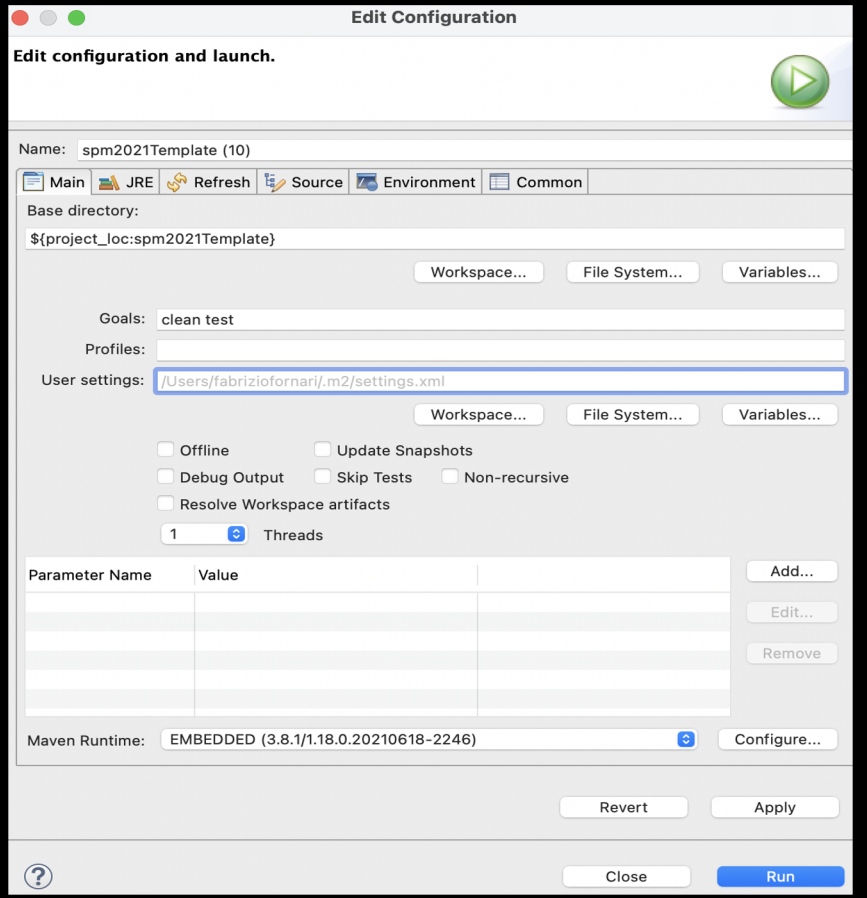




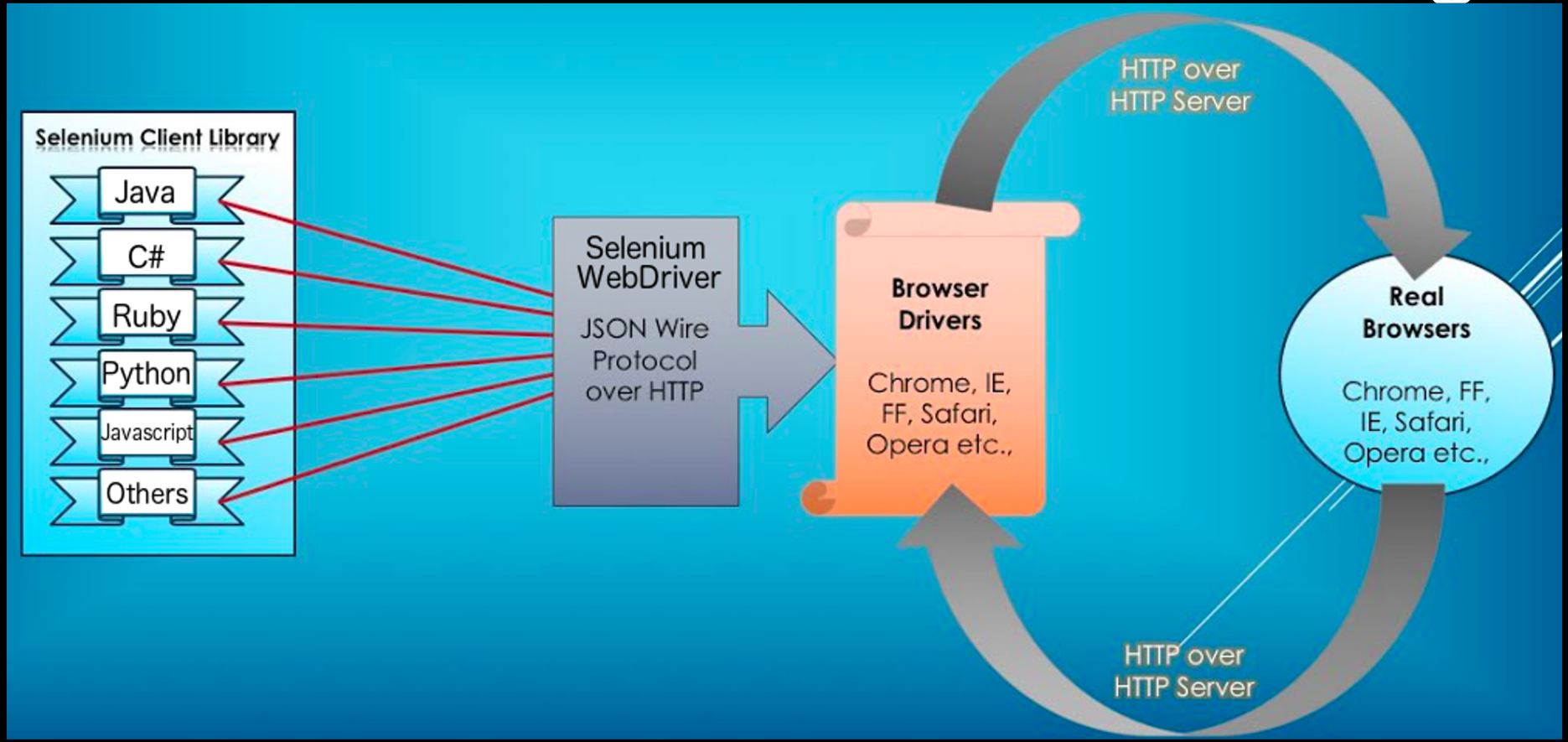
Automated Testing with Maven



clean test

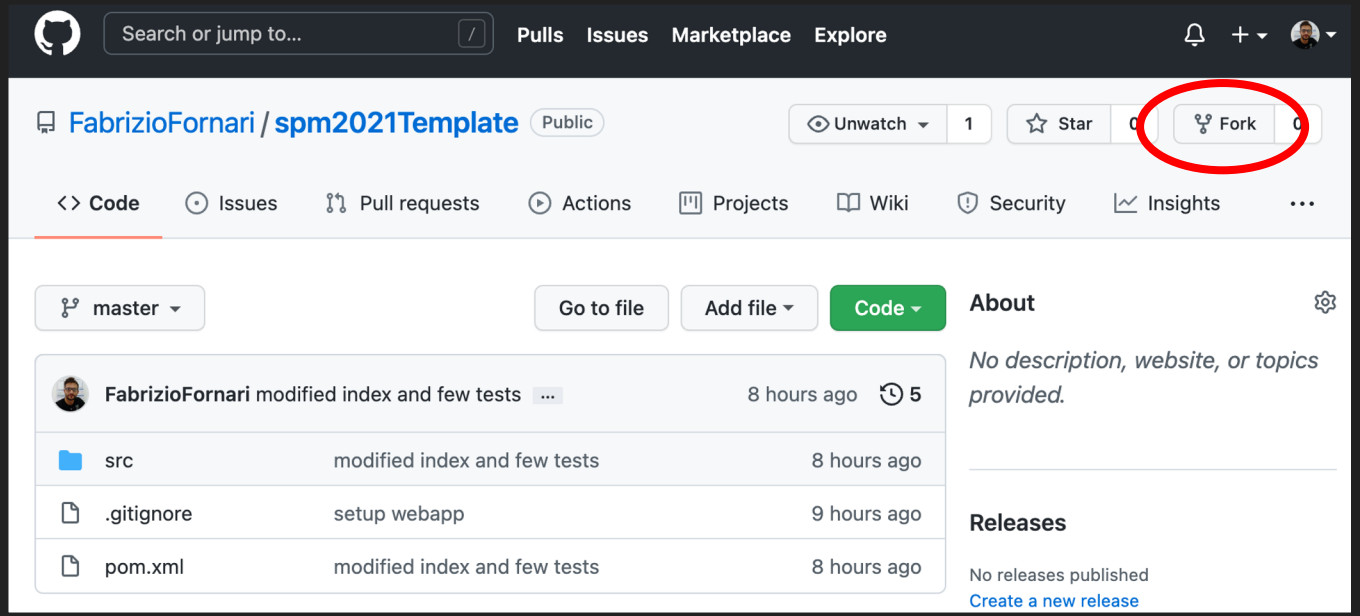


Selenium Automated Web Testing



Fork a GitHub Repository

1. Go to: https://github.com/FabrizioFornari/spm2021Template
2. Fork the repository (which means copy it into your GitHub account)



Fork a GitHub Repository

1. Go to: https://github.com/FabrizioFornari/spm2021Template
2. Fork the repository (which means copy it into your GitHub account)
3. Clone the *smp2021Template* repository that is associated with your GitHub account
4. Import the project as a Maven project into your Development Environment
5. Replace the content of *SeleniumTest.java* with the code you wrote in the previous lecture
6. Run a Selenium test to check that everything works fine

Fork a GitHub Repository

1. Go to: https://github.com/FabrizioFornari/spm2021Template
2. Fork the repository (which means copy it into your GitHub account)
3. Clone the *smp2021Template* repository that is associated with your GitHub account
4. Import the project as a Maven project into your Development Environment
5. Replace the content of *SeleniumTest.java* with the code you wrote in the previous lecture
6. Run a Selenium test to check that everything works fine
7. For this exercise disable the tests that are failing
8. Build the project (Right click on the project root folder, Click *Run As*, Click *Maven Build..*, Write *clean install* in the Goals section, Click *Run*)
9. Commit and push your changes to GitHub

Recap

In project development, normally a deployment process consists of the following steps:

- Write code and test (or vice-versa)
- Build an test the application
- Store the build output as a WAR file
- Deploy the file to a site

Apache Tomcat

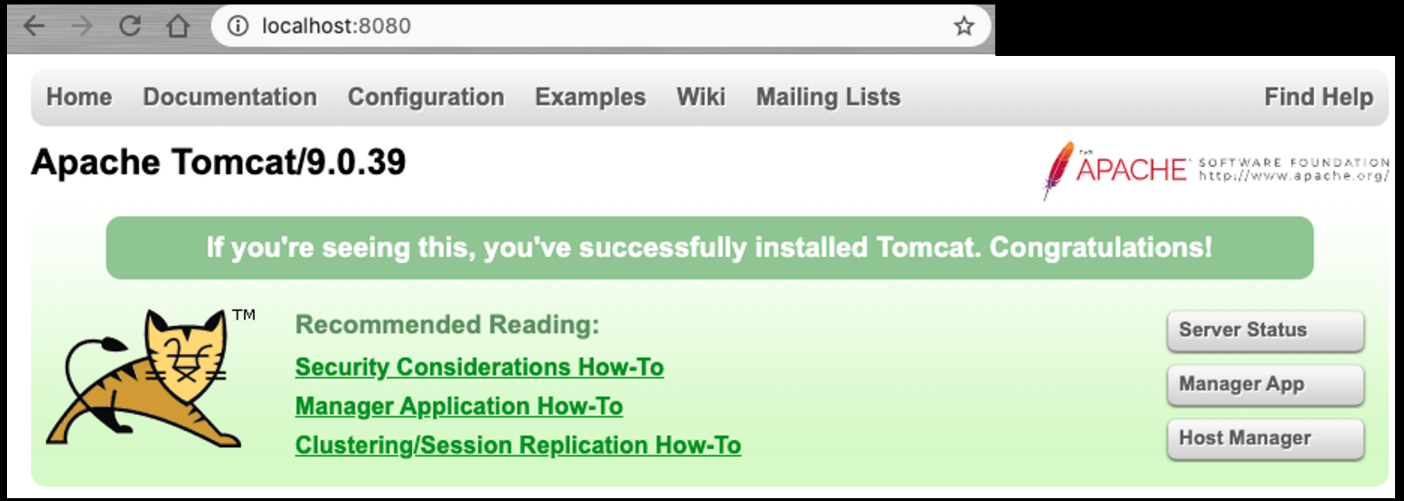
The Apache Tomcat® software is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies. It is a HTTP web server environment in which Java code can run.

http://tomcat.apache.org/

Download Tomcat
https://tomcat.apache.org/download-90.cgi

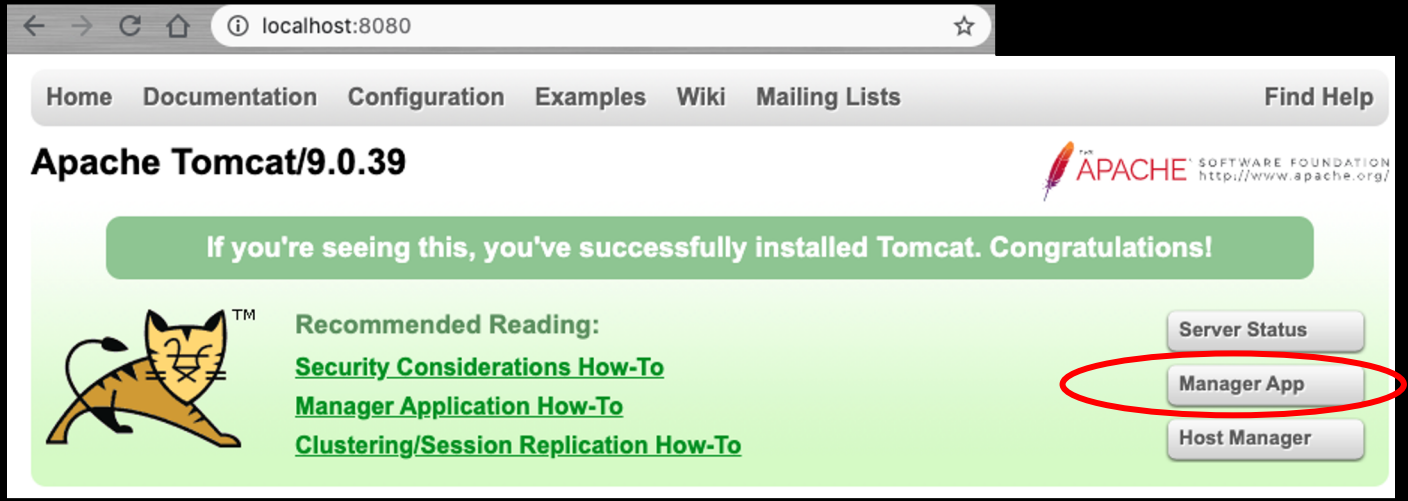
Hands On

- Launch Tomcat
- Navigate the Tomcat folder until the folder bin, then run
  i. ./startup.sh or sudo ./startup.sh
  ii. On windows just type ./startup.bat
- Tomcat is accessible at localhost:8080



Hands On

- Configure Tomcat Manager by clicking on Manager App



Hands On

- Follow the instructions

401 Unauthorized

You are not authorized to view this page. If you have not changed any configuration files, please examine the file `conf/tomcat-users.xml` in your installation. That file must contain the credentials to let you use this webapp.

For example, to add the `manager-gui` role to a user named `tomcat` with a password of `s3cret`, add the following to the config file listed above.

```
<role rolename="manager-gui"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
```

Note that for Tomcat 7 onwards, the roles required to use the manager application were changed from the single `manager` role to the following four roles. You will need to assign the role(s) required for the functionality you wish to access.

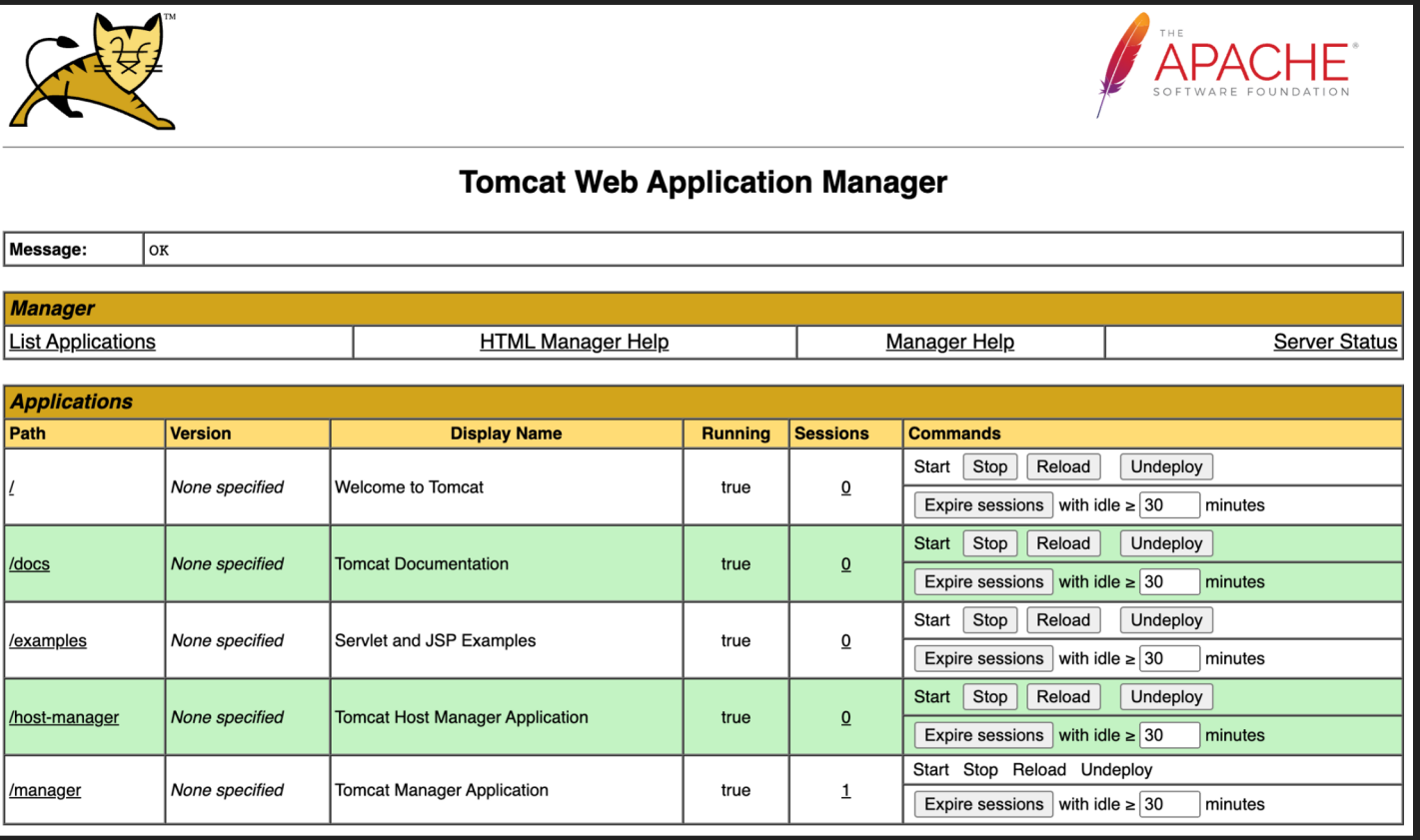
- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

The HTML interface is protected against CSRF but the text and JMX interfaces are not. To maintain the CSRF protection:

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for tools not humans) then the browser must be closed afterwards to terminate the session.

For more information - please see the Manager App How-To.

Tomcat Manager



Tomcat Web Application Manager

Message:	OK

Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

Applications

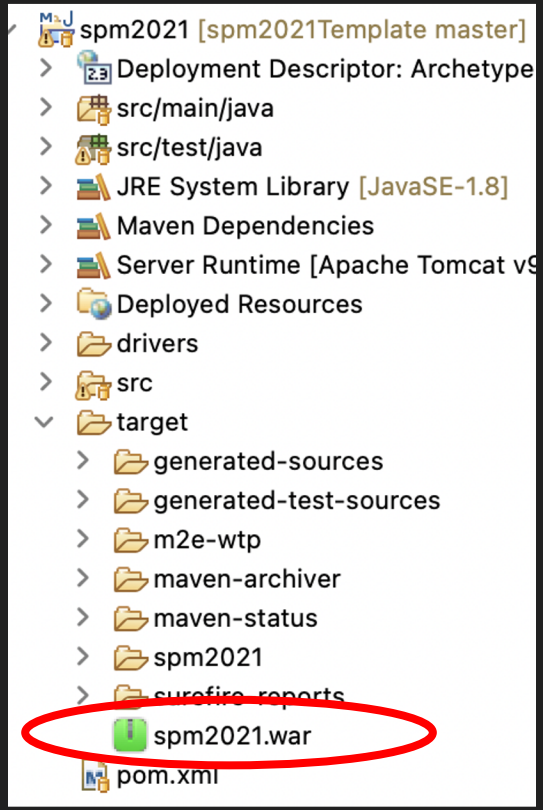
Path	Version	Display Name	Running	Sessions	Commands
/	*None specified*	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	*None specified*	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	*None specified*	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

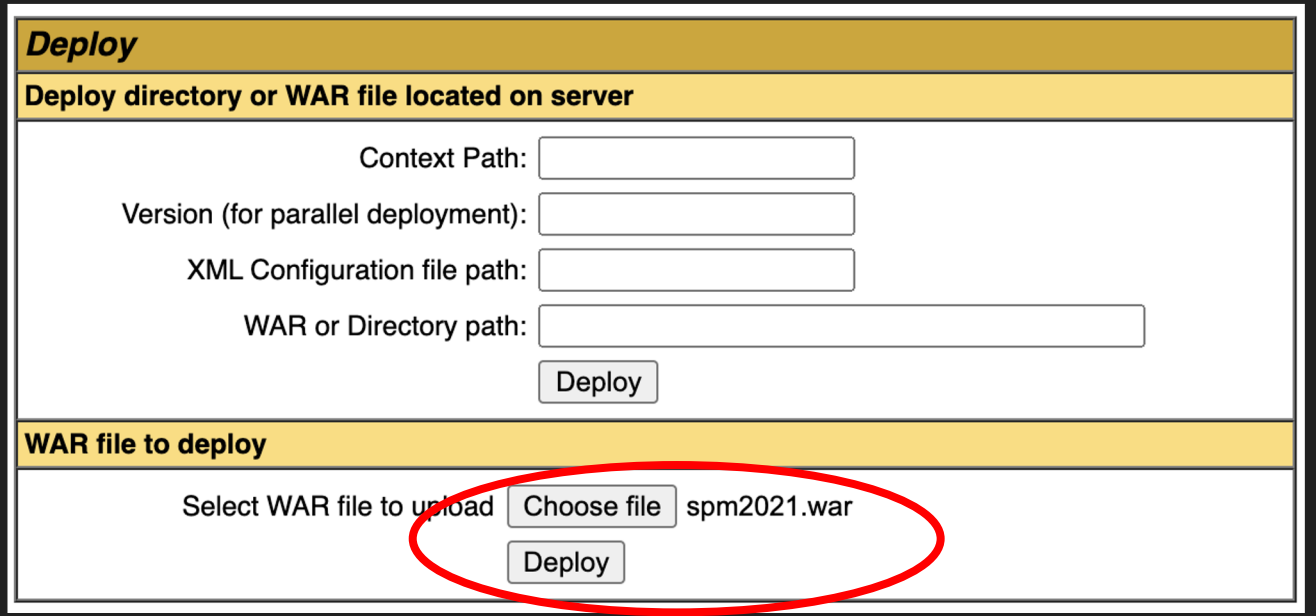
Tomcat Manager

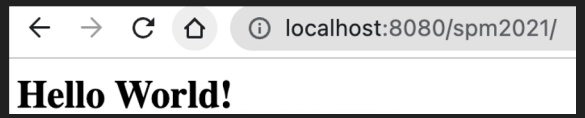
tomcat/conf/tomcat-users.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!--…
-->
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
<role rolename="manager-gui"/>
<user username="tomcat" password="tomcat" roles="manager-gui"/>
</tomcat-users>
```

Tomcat Manager







Selenium Test

Write an run a Selenium Test within *SeleniumTest.java* that Asserts that the title of the web page is actually “SPM 2021”;

Selenium Test

Write and run a Selenium Test within *SeleniumTest.java* that Asserts that the title of the web page is actually “SPM 2021”;

```
@Test
    void checkThisWebAppTitle() throws InterruptedException {

            driver.get("http://localhost:8080/spm2021/");

            Thread.sleep(3000);

            String at = driver.getTitle();
            String et = "SPM 2021";

            //System.out.println(at);
            Thread.sleep(4000);

            Assert.assertEquals(et,at);
    }
```

Second Part

Environments

NOTE: Referred also as Development, Testing, Acceptance and Production (DTAP)

Development

Development and Unit testing for the developed feature are done on the individual developer's laptop or desktop system with a proper version control system in place.

For web based applications, at a minimum, it requires:

- The same web server used in production.
- The same database used in production.
- The same language being used in production.

Build/Test

The build/test server should automatically check out all the code, refresh the database and then execute tests.

All unit tests are run, then integration and regression testing are performed to make sure that all the pieces fit together and nothing previously working was broken.

Staging

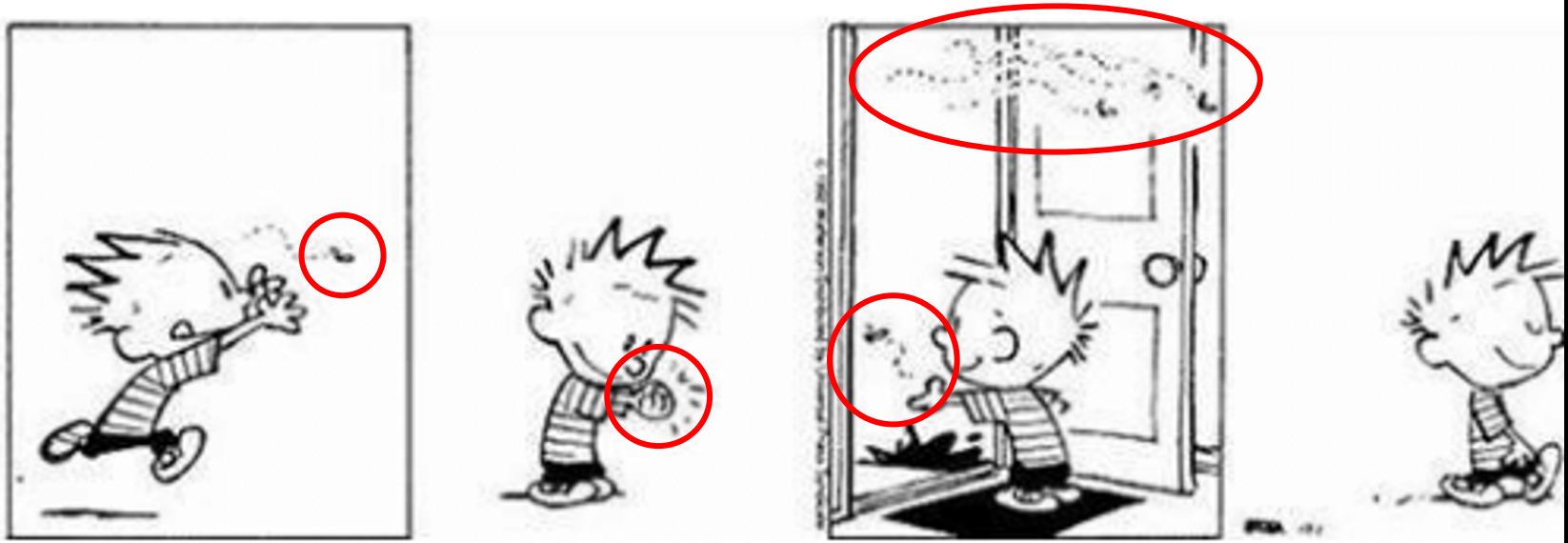
The staging site is used to assemble, test and review new versions of a web app before it goes into production.

It is often used to present the client with the final project for them to perform ***Acceptance testing***

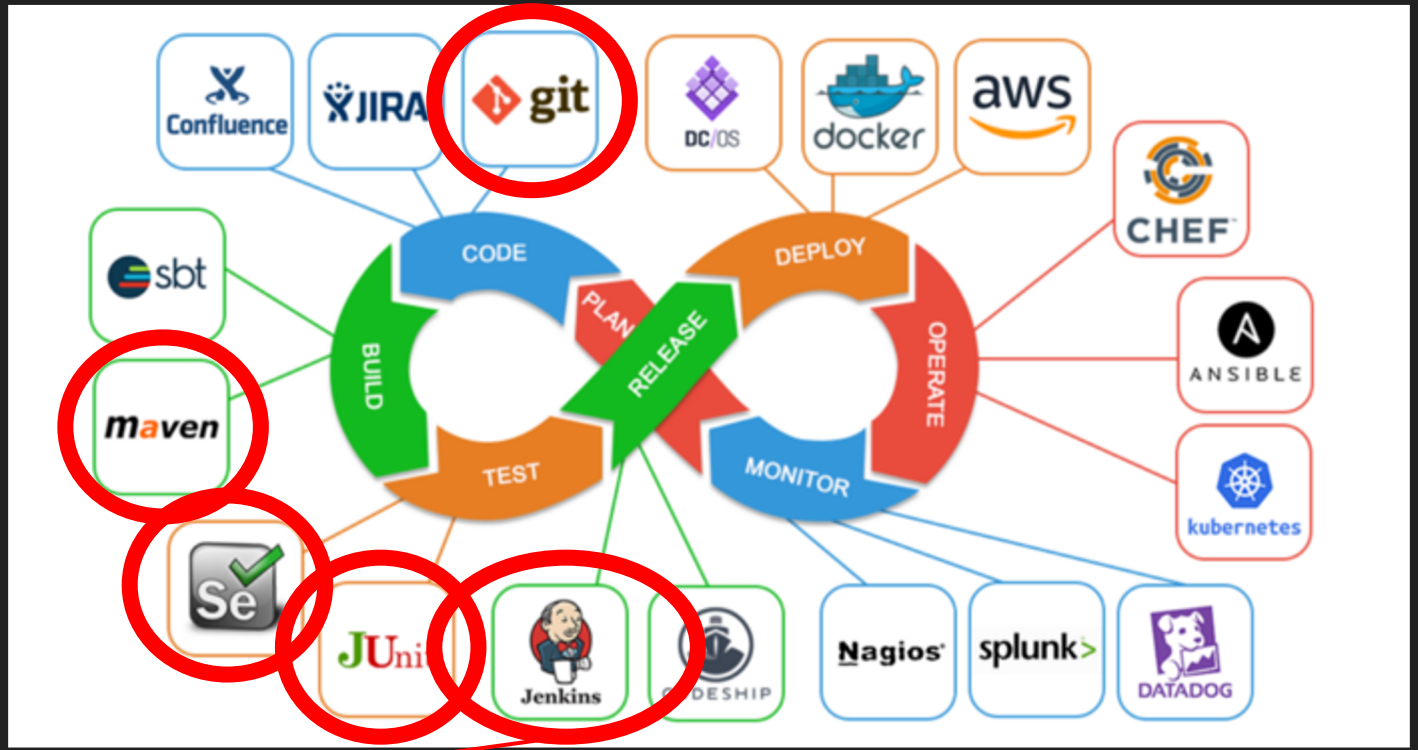
Production

The accepted product, is deployed to a Production environment, making it available to all users of the system.

Regression:
"when you fix one bug, you introduce several newer bugs."

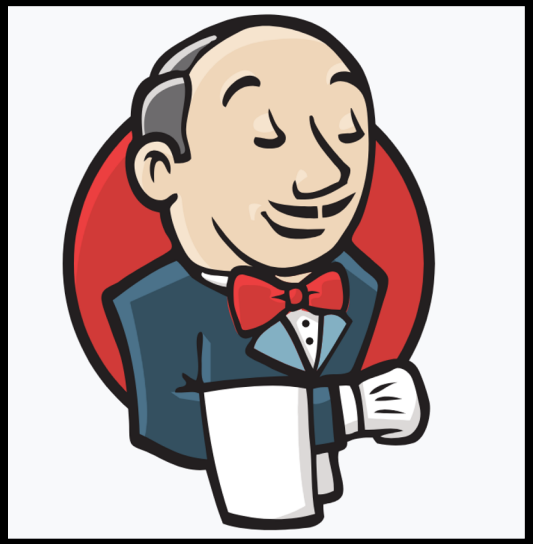


DevOps



Our Focus

Jenkins



Jenkins is used to build and test your product continuously, so developers can continuously integrate changes into the build.

https://jenkins.io/

Continuous Integration

In its simplest form, it involves a tool that monitors your version control system for changes.

Whenever a change is detected, this tool automatically compiles and tests your application.

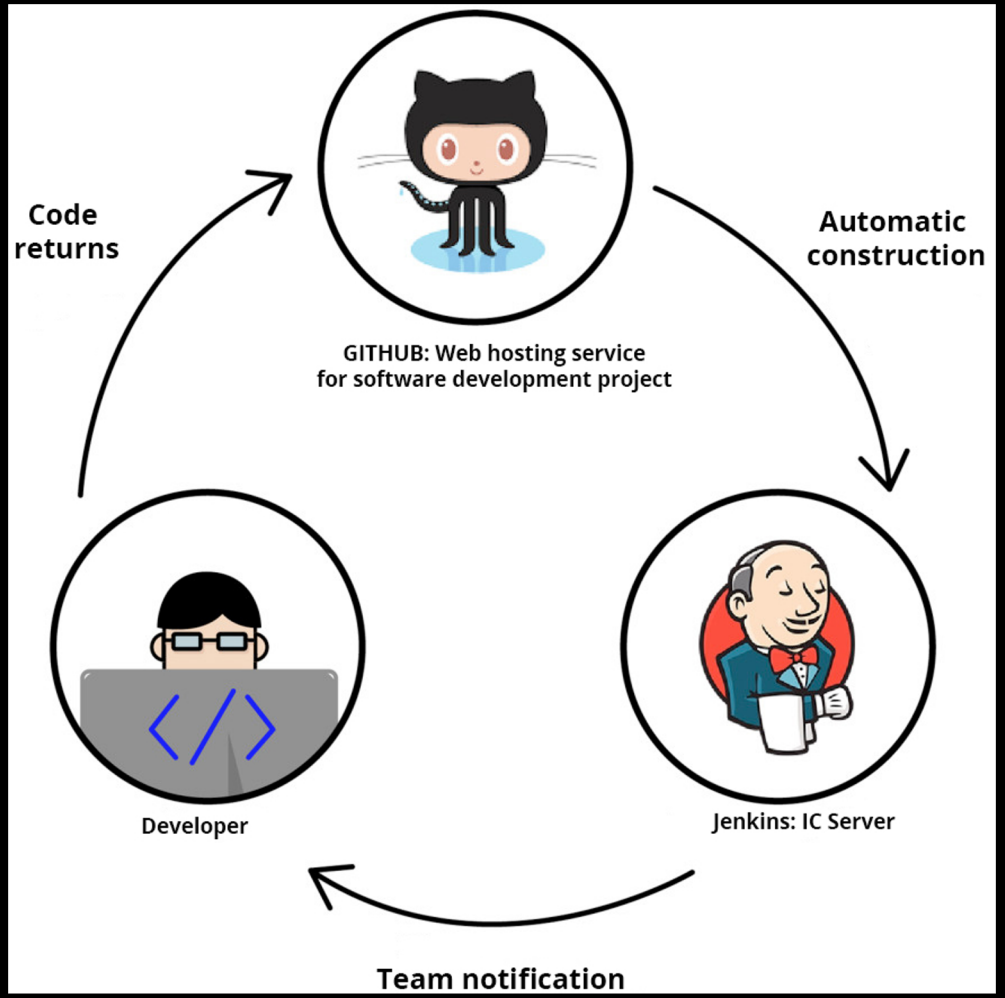
If something goes wrong, the tool immediately notifies the developers so that they can fix the issue immediately.

Why is it needed?

The objectives are: to collaborate, to speed up development, to increase reliability by testing that everything works as expected, especially integrating all the functionalities developed by the various members of the Team!

Continuous Integration with Jenkins

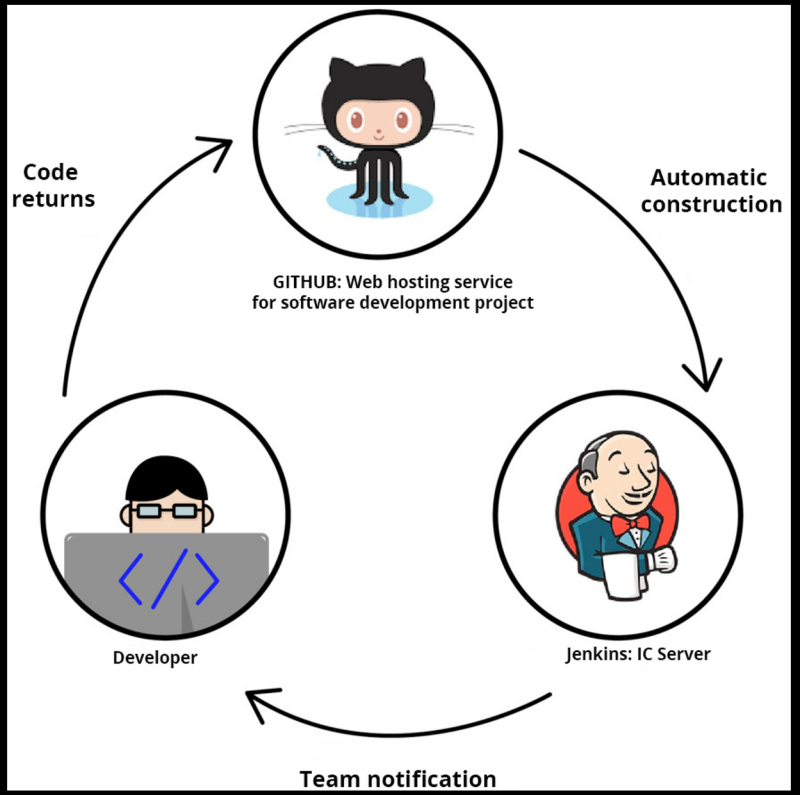
Jenkins triggers a build upon every commit to the source code repository, typically to a development branch.



Continuous Integration with Jenkins: Effects

1. Jenkins can run a unit test suite and can determine which commit caused the build to fail.

2. If all the unit tests pass, then the build pipeline can proceed to the next phase with integration tests which typically take longer to run. (We avoid to run them locally)



These simplified steps encompass the spirit of a continuous integration (CI) environment.

Jenkins

Plugins

https://plugins.jenkins.io/

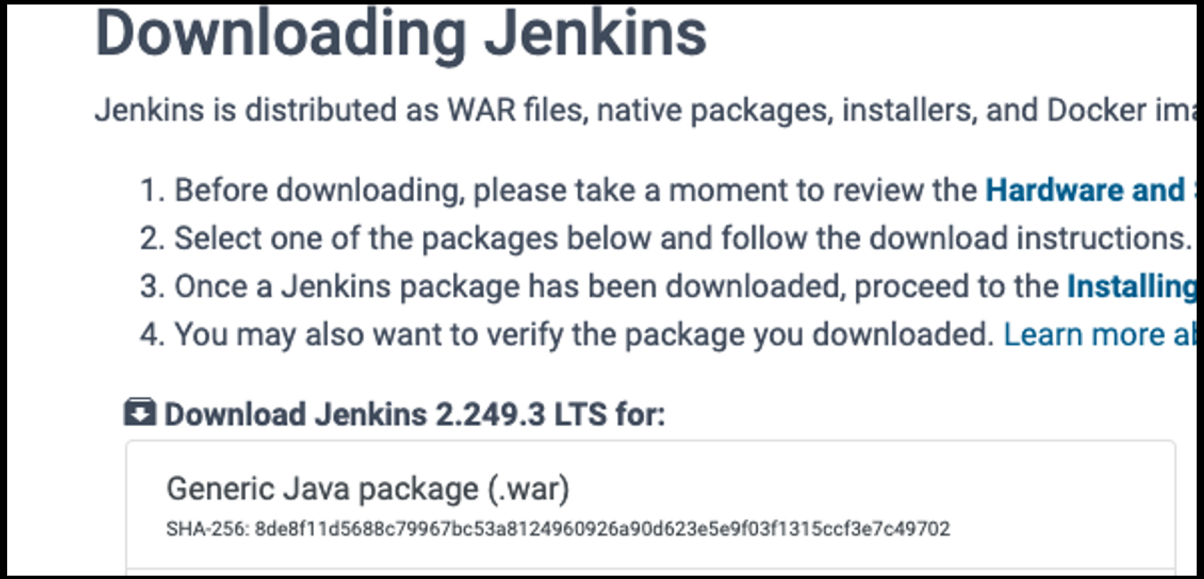
Jenkins on Tomcat

Download the latest .war file
https://www.jenkins.io/download/

Deploy the .war on Tomcat
https://www.jenkins.io/doc/book/installing/#war-file

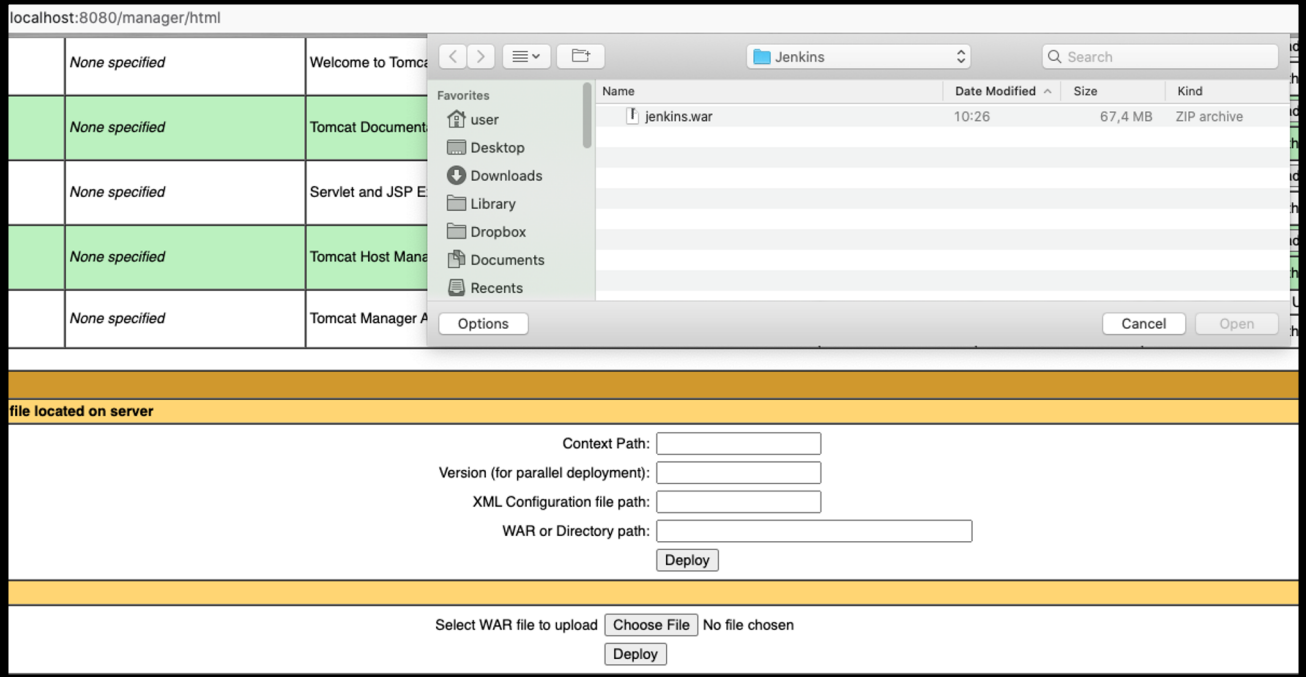
Hands On

- Download Jenkins.war https://www.jenkins.io/download/

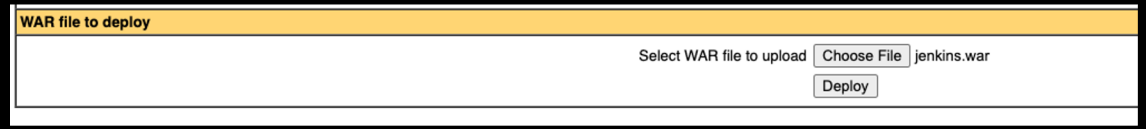


https://www.jenkins.io/doc/book/installing/

Hands On

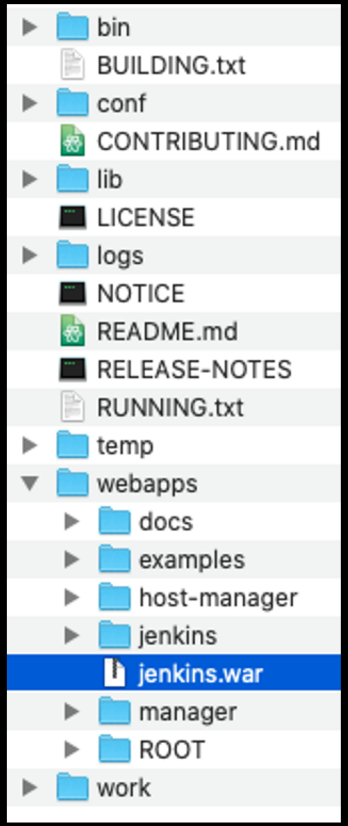


- Deploy Jenkins on Tomcat
- Choose File and select jenkins.war
- Click Deploy



Hands On

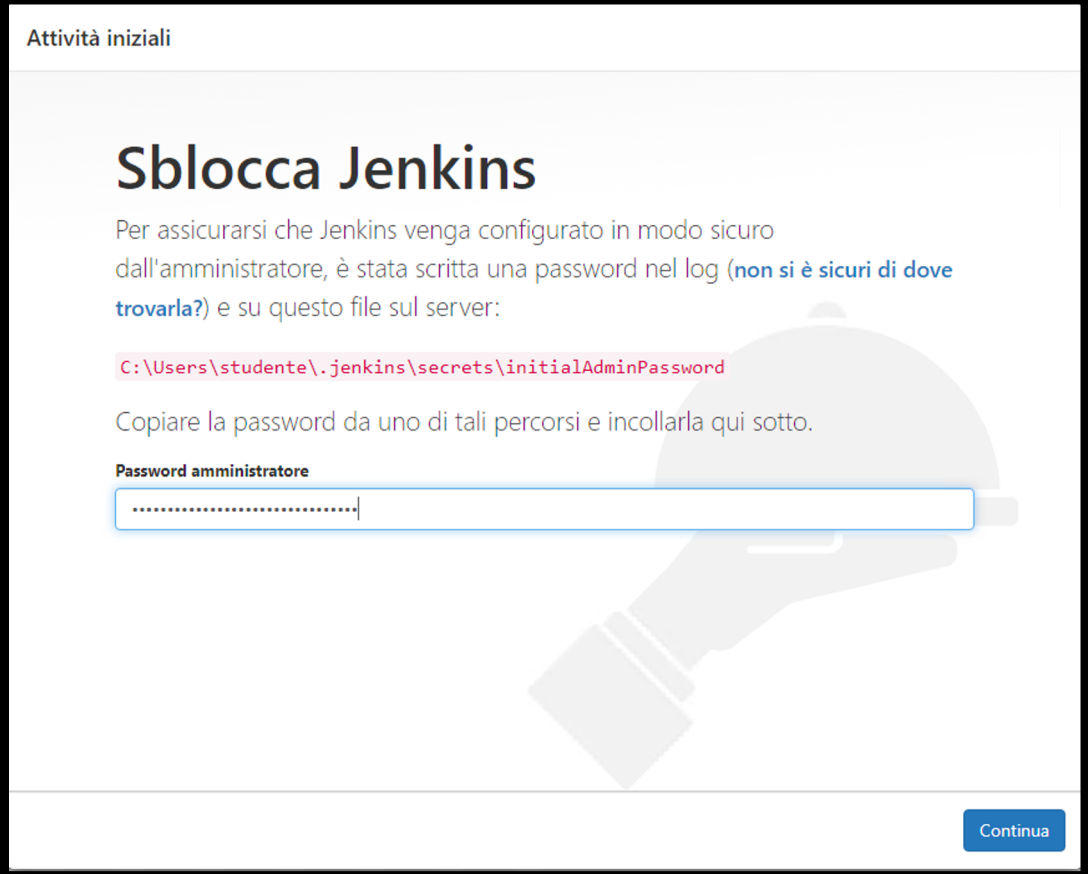
- If the deployment does not work just copy the jenkins.war and paste it under the webapp folder inside tomcat folder



- On tomcat manager you will see /jenkins, press Start if it is not started, then click on /jenkins

Hands On

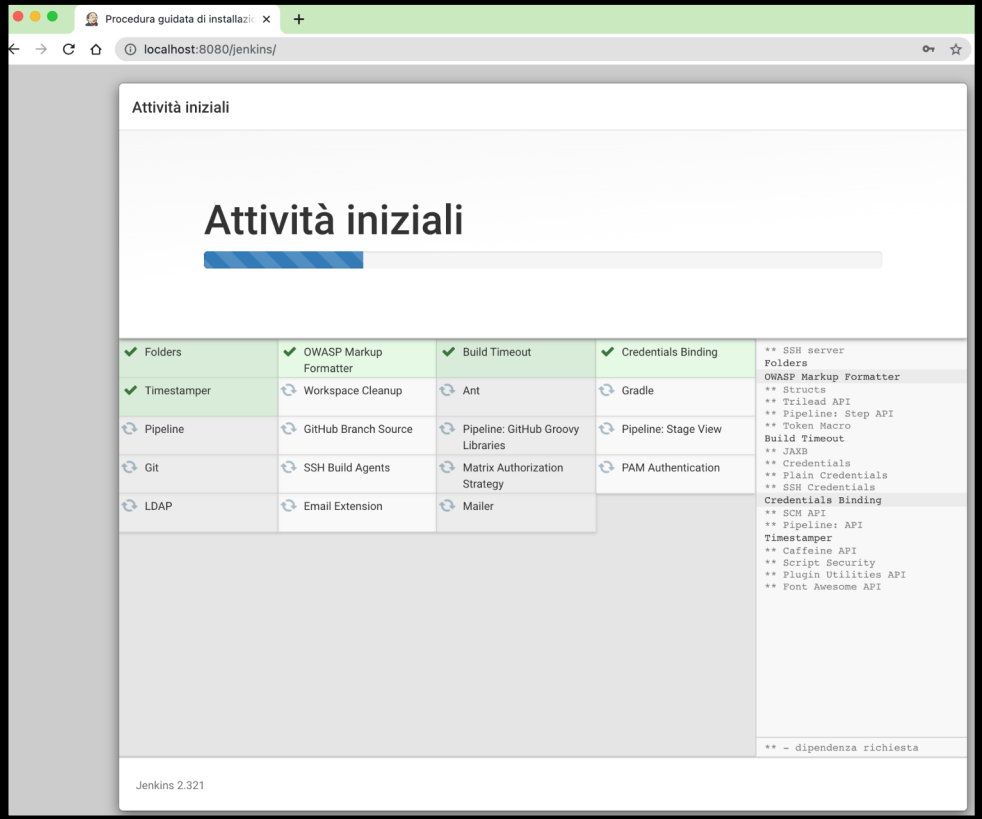
1. Go to the path shown on your screen open the file named initialAdminPassword, copy and paste the password



Hands On

1. Go to the path shown on your screen open the file named initialAdminPassword, copy and paste the password

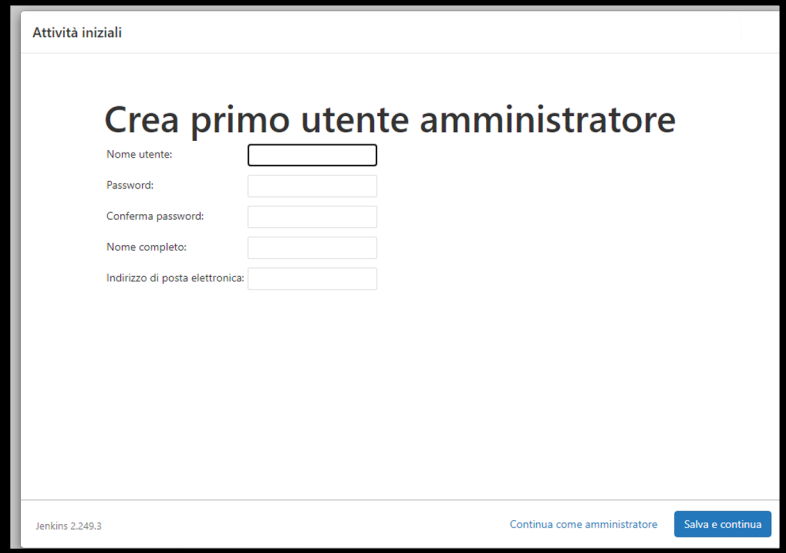
1. Install default plugins



Hands On

1. Create an admin user

http://localhost:8080/jenkins/

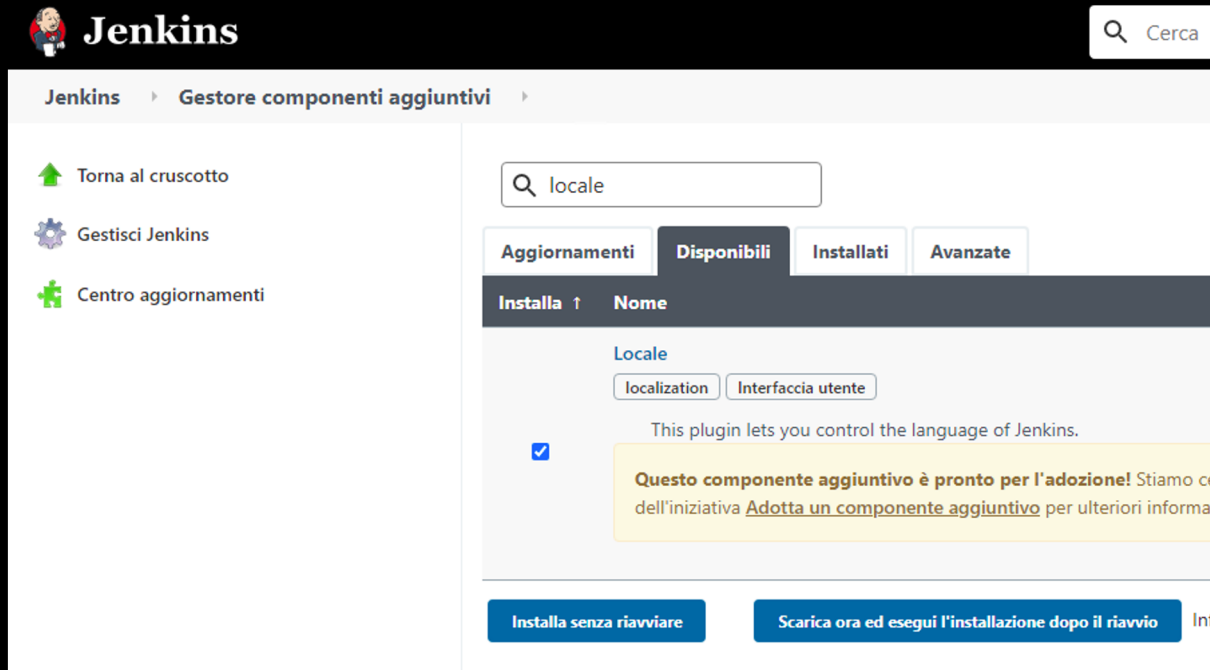


Change UI Language

1. Click Manage Jenkins > Manage Plugins > ['Available' tab]
2. In the Filter, search for: Locale .
3. Click on Locale Plugin checkbox and Install without restart button.

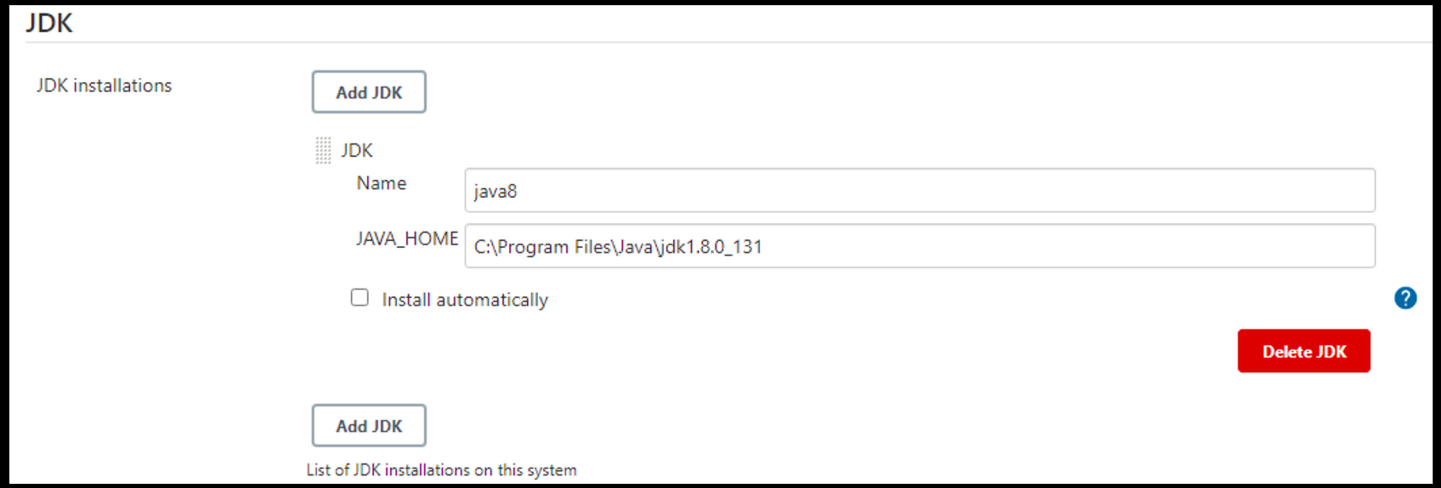
After installation is complete:

- Under Manage Jenkins > Configure System there should be a "Locale" section.
- Enter the default language_LOCALE code for English: en_US
- Click on Ignore browser preference and force this language to all users checkbox.



Configure JDK

Manage Jenkins > Global Tool Configuration

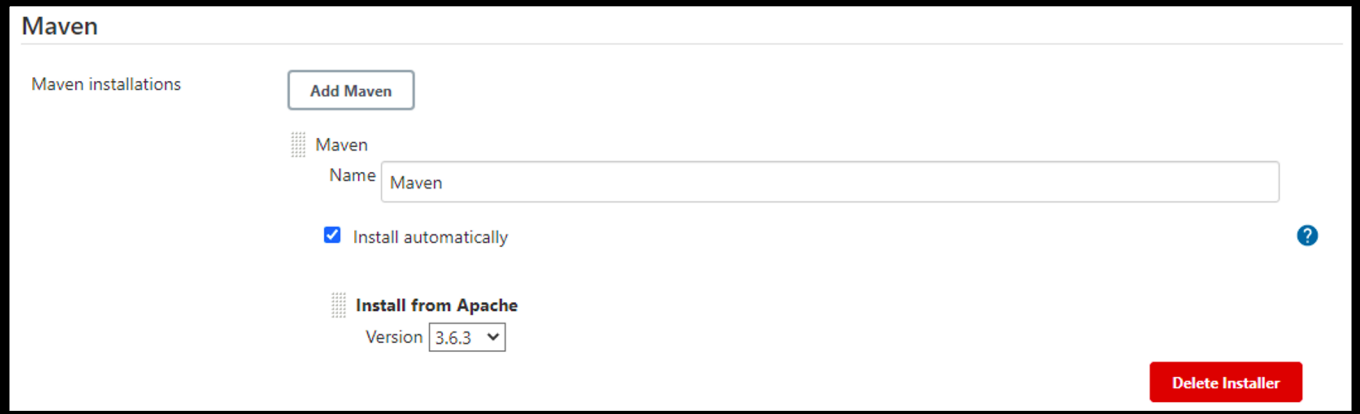


Configure JDK & Maven

Manage Jenkins --> Global Tool Configuration

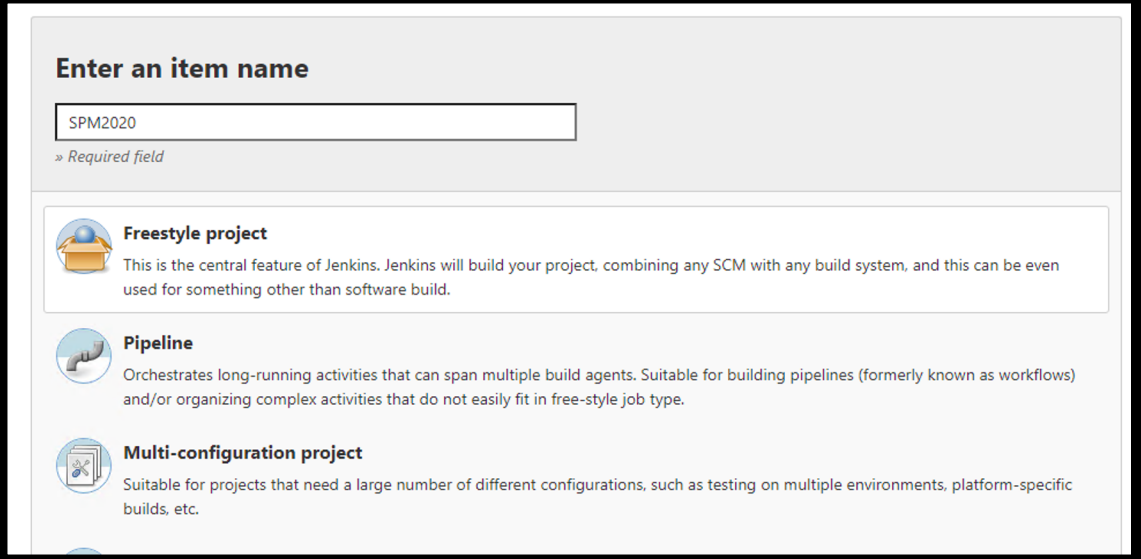
Or

Tell Jenkins to install a version

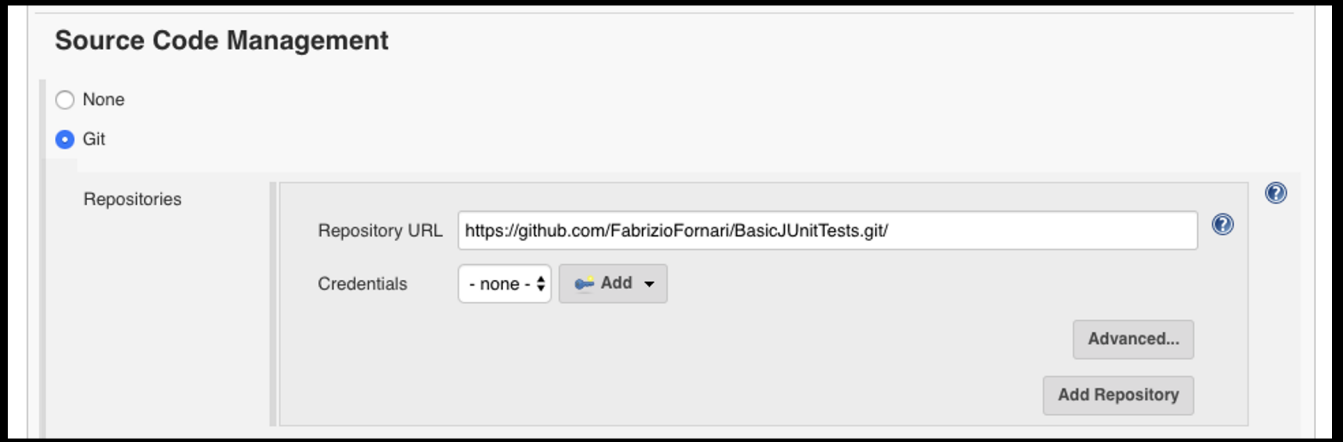


Create Job

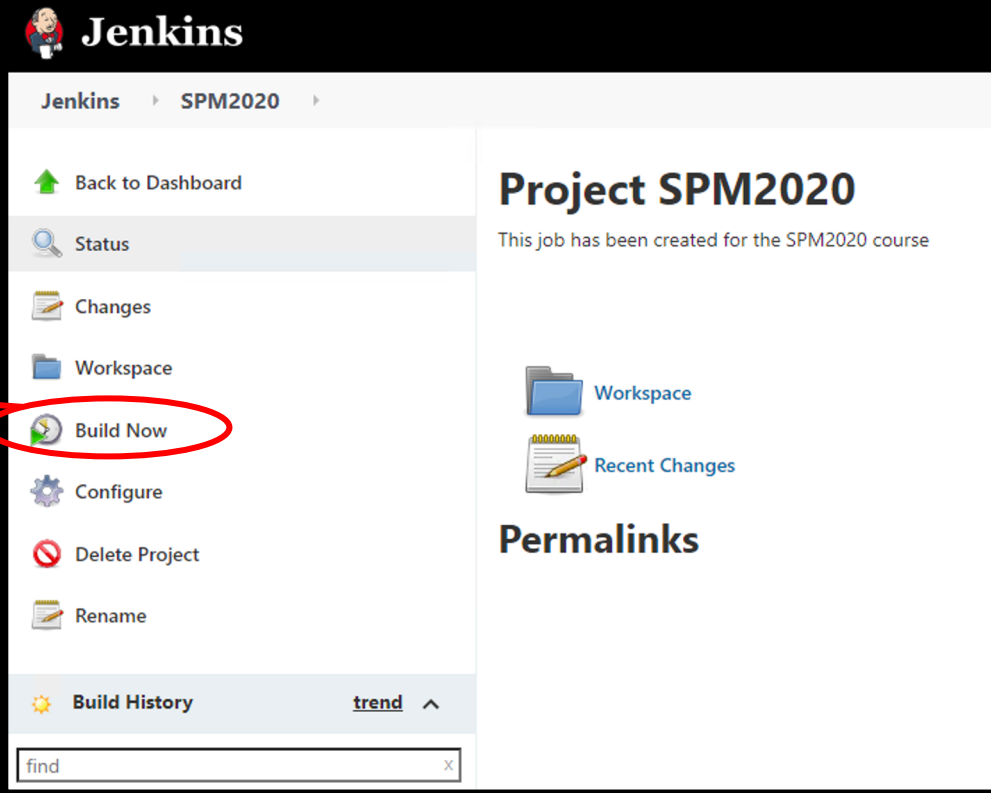
1 Create a new Job



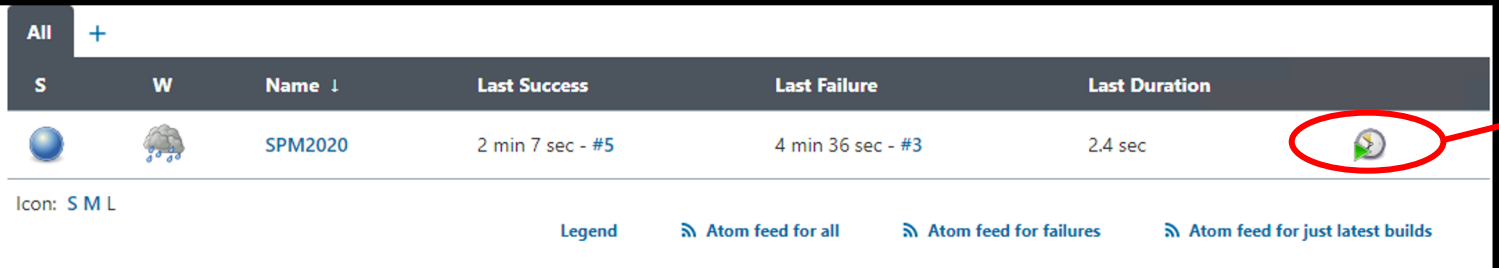
2 Link it with your GitHub Repository



Run a Job



Run a Job



Run a Job