

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.



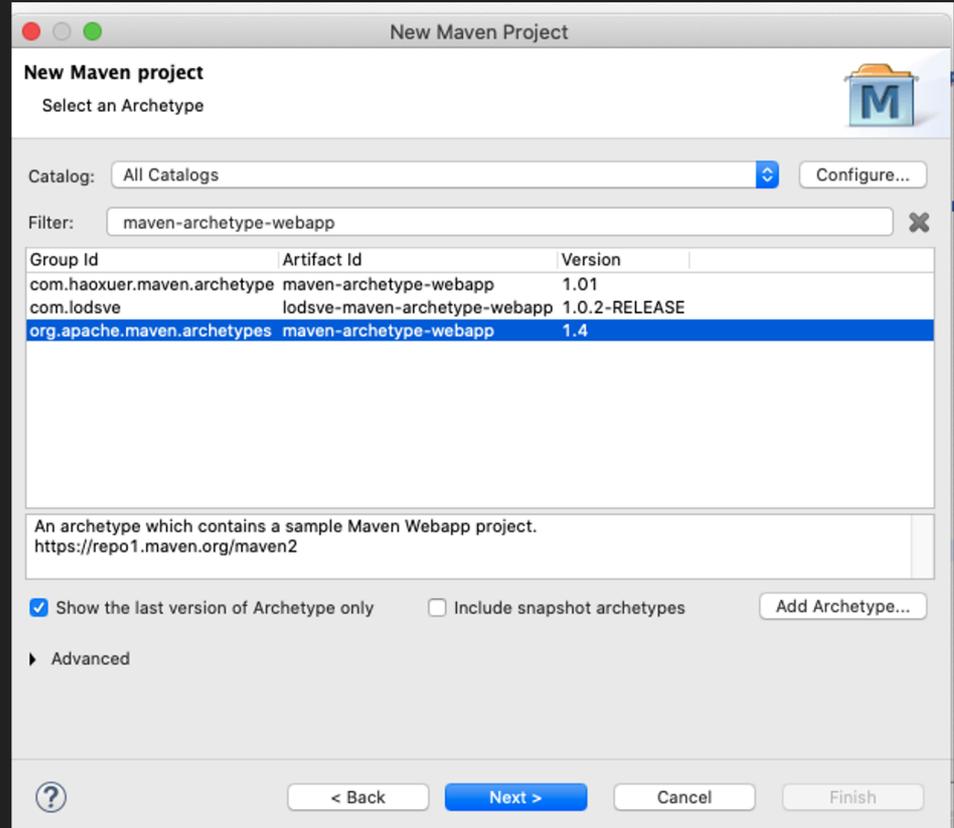
<http://maven.apache.org/>

Maven - Archetypes

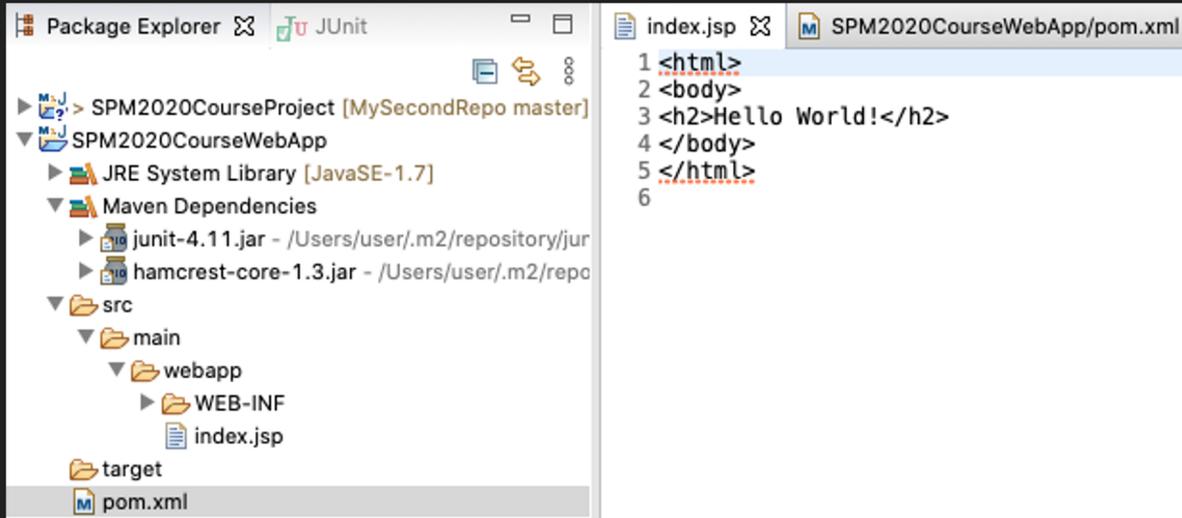
Maven archetypes are project templates that allow users to generate new projects easily

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

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



Maven - Archetype WebApp



The screenshot displays an IDE interface with two main panels. The left panel, titled 'Package Explorer', shows a project structure for 'SPM2020CourseWebApp'. It includes a 'Maven Dependencies' section with 'junit-4.11.jar' and 'hamcrest-core-1.3.jar'. The 'src' directory contains a 'main' folder, which in turn contains a 'webapp' folder. Inside 'webapp', there is a 'WEB-INF' folder and an 'index.jsp' file. The right panel shows the content of 'index.jsp', which contains the following HTML code:

```
1 <html>
2 <body>
3 <h2>Hello World!</h2>
4 </body>
5 </html>
6
```

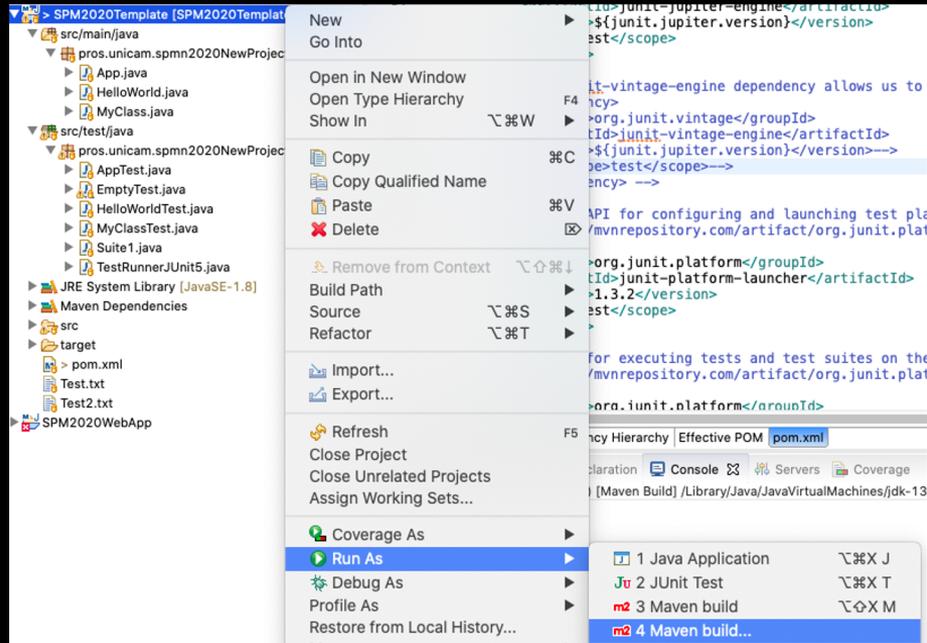
Manual Testing in Eclipse

The screenshot displays the Eclipse IDE interface during a manual test run. On the left, the Package Explorer shows a test suite named 'HelloWorldTest' with several test methods, including 'testHelloShouldReturnAString()'. The main editor shows the source code of 'App.java' with a red circle highlighting the '@Disabled' annotation on line 57. A context menu is open over the code, with 'Run As' selected. The Console at the bottom shows the execution output of the test, including the message 'HelloWorldTest (1) [JUnit] /Library/Java/JavaVirtualMachines/...'. The console output includes the following text:

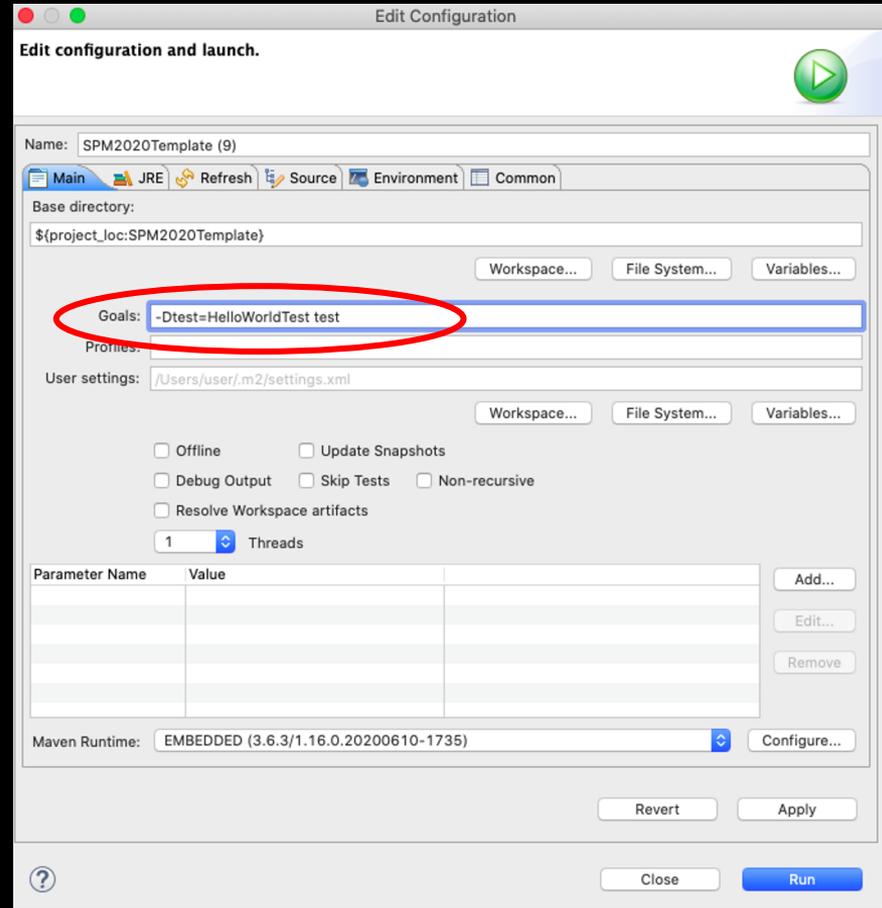
```
<terminated> HelloWorldTest (1) [JUnit] /Library/Java/JavaVirtualMachines/...
Nov 10, 2020 11:34:48 AM pros.unicam.spmn2020Newl
INFO: @BeforeEach - executes before each test me
Nov 10, 2020 11:34:48 AM pros.unicam.spmn2020Newl
INFO: @AfterEach - executes after each test metho
Nov 10, 2020 11:34:48 AM pros.unicam.spmn2020Newl
INFO: @AfterAll - executes once after all test m
```

Manual Testing with Maven

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

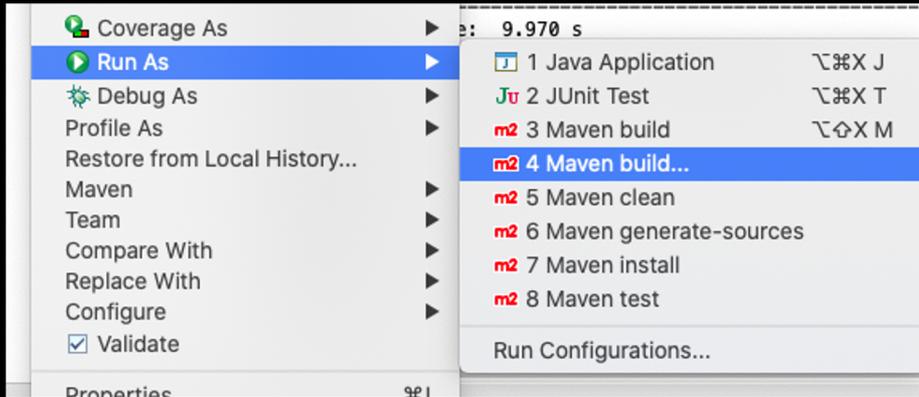


The screenshot shows an IDE interface. On the left, a project tree displays a project named 'SPM2020Template'. The 'src/test/java' directory is expanded, showing several test classes including 'HelloWorldTest.java'. A context menu is open over the 'pom.xml' file, with the 'Run As' option selected. A sub-menu is visible, listing options: '1 Java Application', '2 JUnit Test', '3 Maven build', and '4 Maven build...'. The '4 Maven build...' option is highlighted.

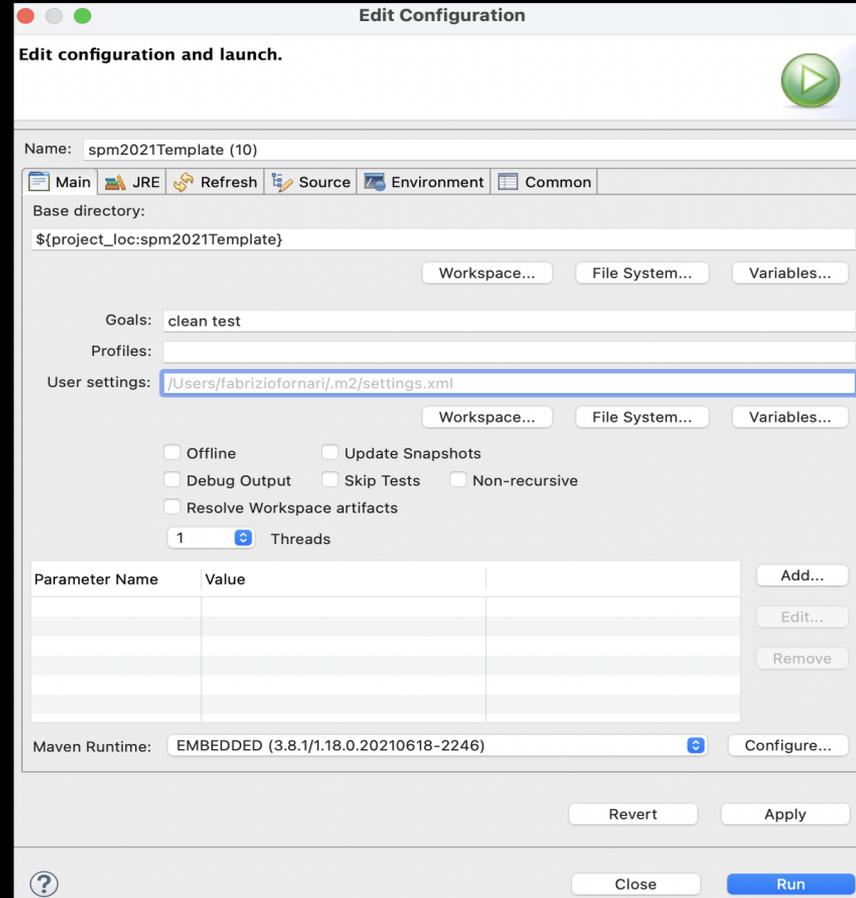


The screenshot shows the 'Edit Configuration' dialog box. The 'Name' field is 'SPM2020Template (9)'. The 'Base directory' is '\$(project_loc:SPM2020Template)'. The 'Goals' field is highlighted with a red circle and contains the text '-Dtest=HelloWorldTest test'. The 'User settings' field is '/Users/user/m2/settings.xml'. The 'Maven Runtime' is 'EMBEDDED (3.6.3/1.16.0.20200610-1735)'. The 'Run' button is highlighted in blue.

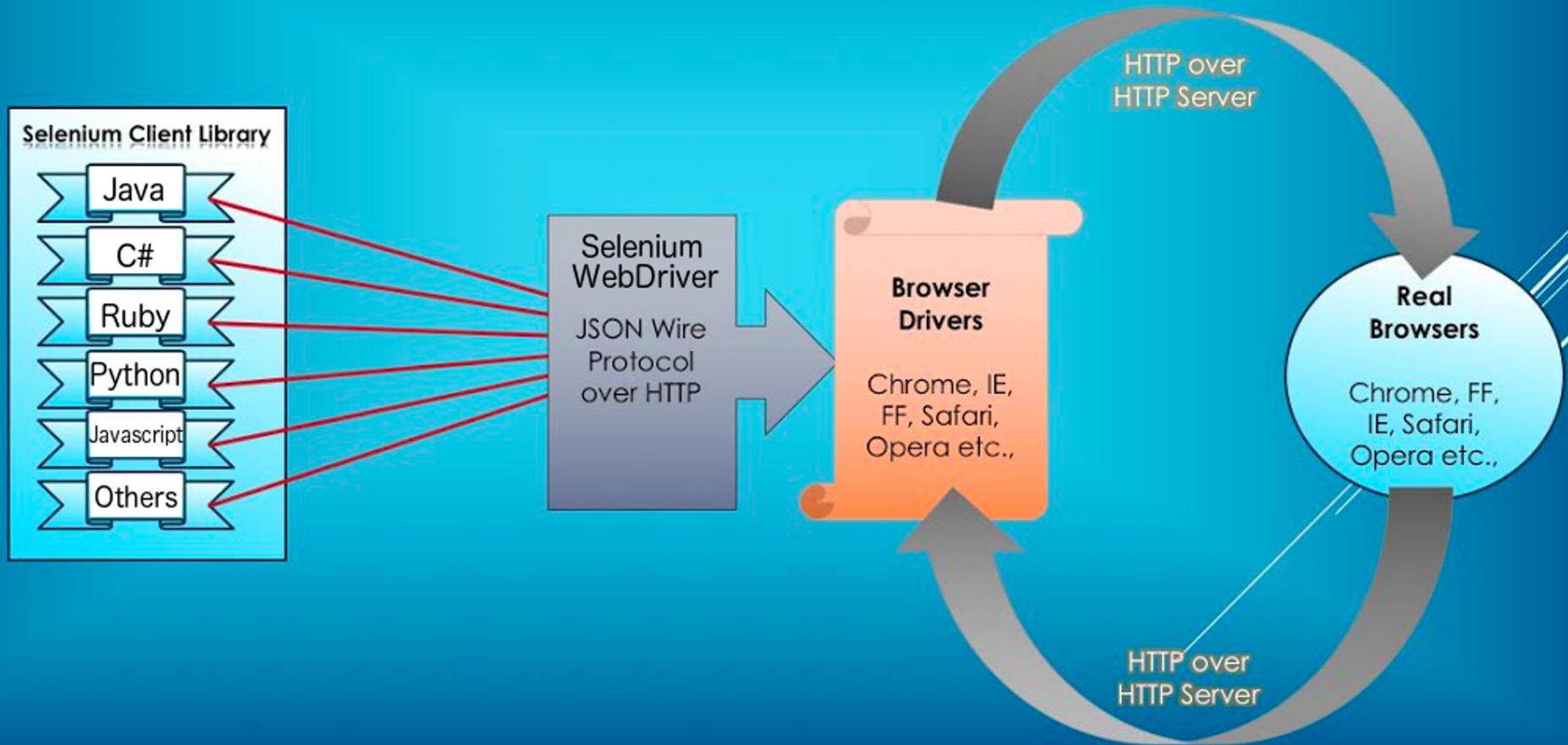
Automated Testing with Maven



clean test

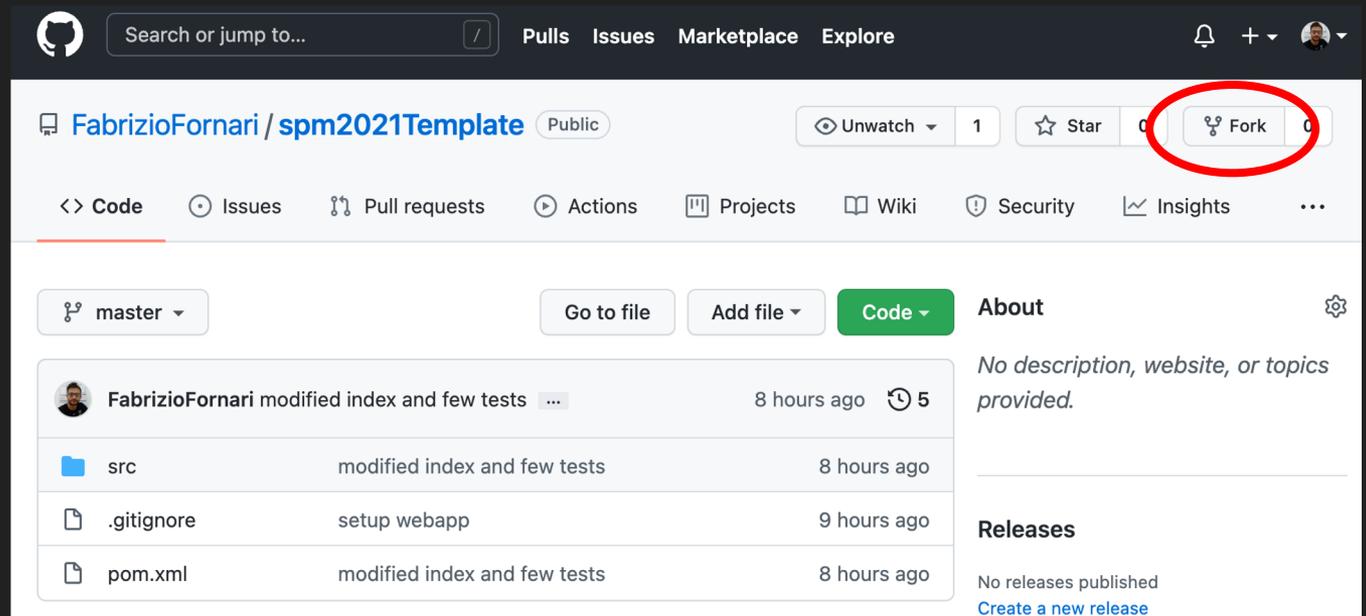


Selenium Automated Web Testing



Fork a GitHub Repository

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



The screenshot shows the GitHub interface for the repository `FabrizioFornari / spm2021Template`. The repository is public. The `Fork` button is circled in red. The repository has 1 star and 0 forks. The `Code` button is highlighted in green. The repository description is "No description, website, or topics provided." The repository has 5 releases. The repository files are:

File	Commit Message	Time
src	modified index and few tests	8 hours ago
.gitignore	setup webapp	9 hours ago
pom.xml	modified index and few tests	8 hours ago

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 *spm2021Template* 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 *spm2021Template* 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 and 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.**

Download Tomcat

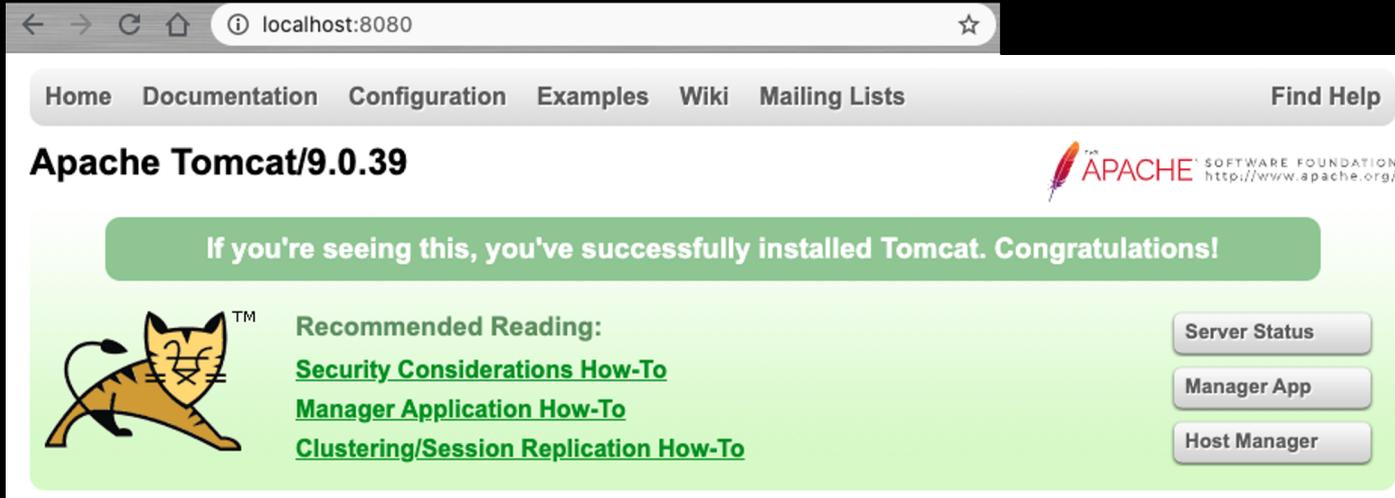
<https://tomcat.apache.org/download-90.cgi>



<http://tomcat.apache.org/>

Hands On

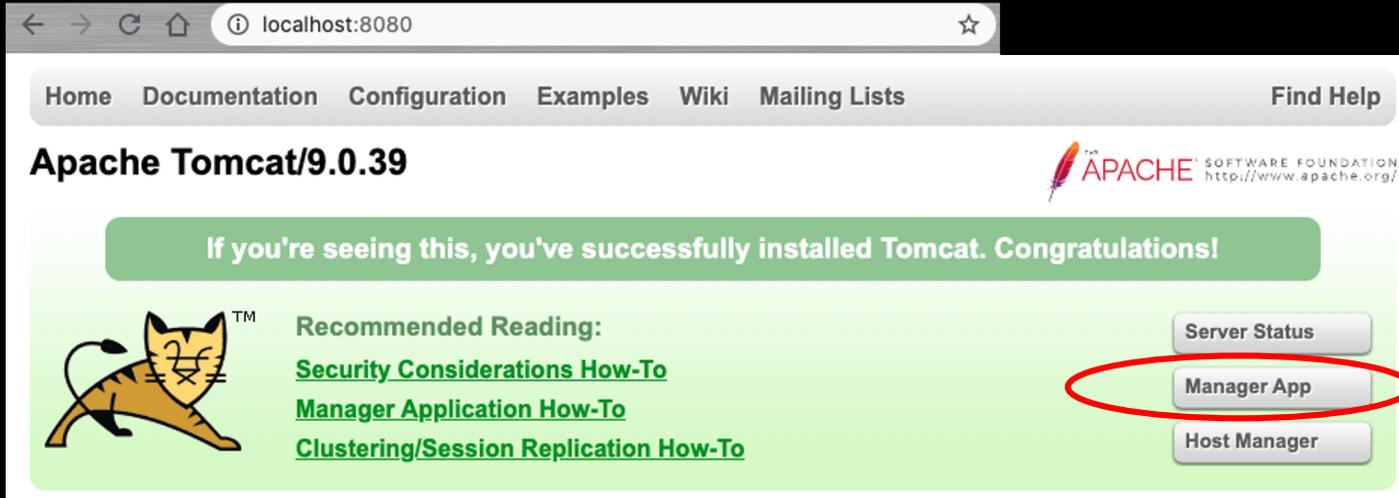
- Launch Tomcat
- Navigate the Tomcat folder until the folder bin, then run
 - i. `./startup.sh` or `sudo ./startup.sh`
 - ii. On windows just type `./startup.bat`
- Tomcat is accessible at localhost:8080



The screenshot shows a web browser window with the address bar set to `localhost:8080`. The page title is "Apache Tomcat/9.0.39". The navigation menu includes "Home", "Documentation", "Configuration", "Examples", "Wiki", "Mailing Lists", and "Find Help". The Apache Software Foundation logo is visible in the top right. A green banner displays the message: "If you're seeing this, you've successfully installed Tomcat. Congratulations!". Below this, there is a "Recommended Reading" section with three links: [Security Considerations How-To](#), [Manager Application How-To](#), and [Clustering/Session Replication How-To](#). To the left of these links is the Tomcat logo, a stylized orange and black striped cat. On the right side, there are three buttons: "Server Status", "Manager App", and "Host Manager".

Hands On

- Configure Tomcat Manager by clicking on Manager App



localhost:8080

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.39

APACHE SOFTWARE FOUNDATION
http://www.apache.org/

If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Server Status

Manager App

Host Manager

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 <input type="button" value="Stop"/> <input type="button" value="Reload"/> <input type="button" value="Undeploy"/> <input type="button" value="Expire sessions"/> with idle \geq <input type="text" value="30"/> minutes
/docs	None specified	Tomcat Documentation	true	0	Start <input type="button" value="Stop"/> <input type="button" value="Reload"/> <input type="button" value="Undeploy"/> <input type="button" value="Expire sessions"/> with idle \geq <input type="text" value="30"/> minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start <input type="button" value="Stop"/> <input type="button" value="Reload"/> <input type="button" value="Undeploy"/> <input type="button" value="Expire sessions"/> with idle \geq <input type="text" value="30"/> minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start <input type="button" value="Stop"/> <input type="button" value="Reload"/> <input type="button" value="Undeploy"/> <input type="button" value="Expire sessions"/> with idle \geq <input type="text" value="30"/> minutes
/manager	None specified	Tomcat Manager Application	true	1	Start <input type="button" value="Stop"/> <input type="button" value="Reload"/> <input type="button" value="Undeploy"/> <input type="button" value="Expire sessions"/> with idle \geq <input type="text" value="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

- spm2021 [spm2021Template master]
 - Deployment Descriptor: Archetype
 - src/main/java
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - Server Runtime [Apache Tomcat v9.0.0]
 - Deployed Resources
 - drivers
 - src
 - target
 - generated-sources
 - generated-test-sources
 - m2e-wtp
 - maven-archiver
 - maven-status
 - spm2021
 - surefire-reports
 - spm2021.war**
 - pom.xml

Deploy

Deploy directory or WAR file located on server

Context Path:

Version (for parallel deployment):

XML Configuration file path:

WAR or Directory path:

Deploy

WAR file to deploy

Select WAR file to upload spm2021.war

Deploy

localhost:8080/spm2021/

Hello World!

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**



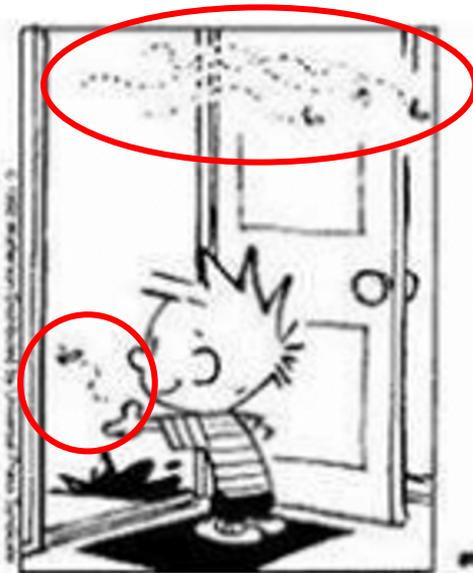
Production

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

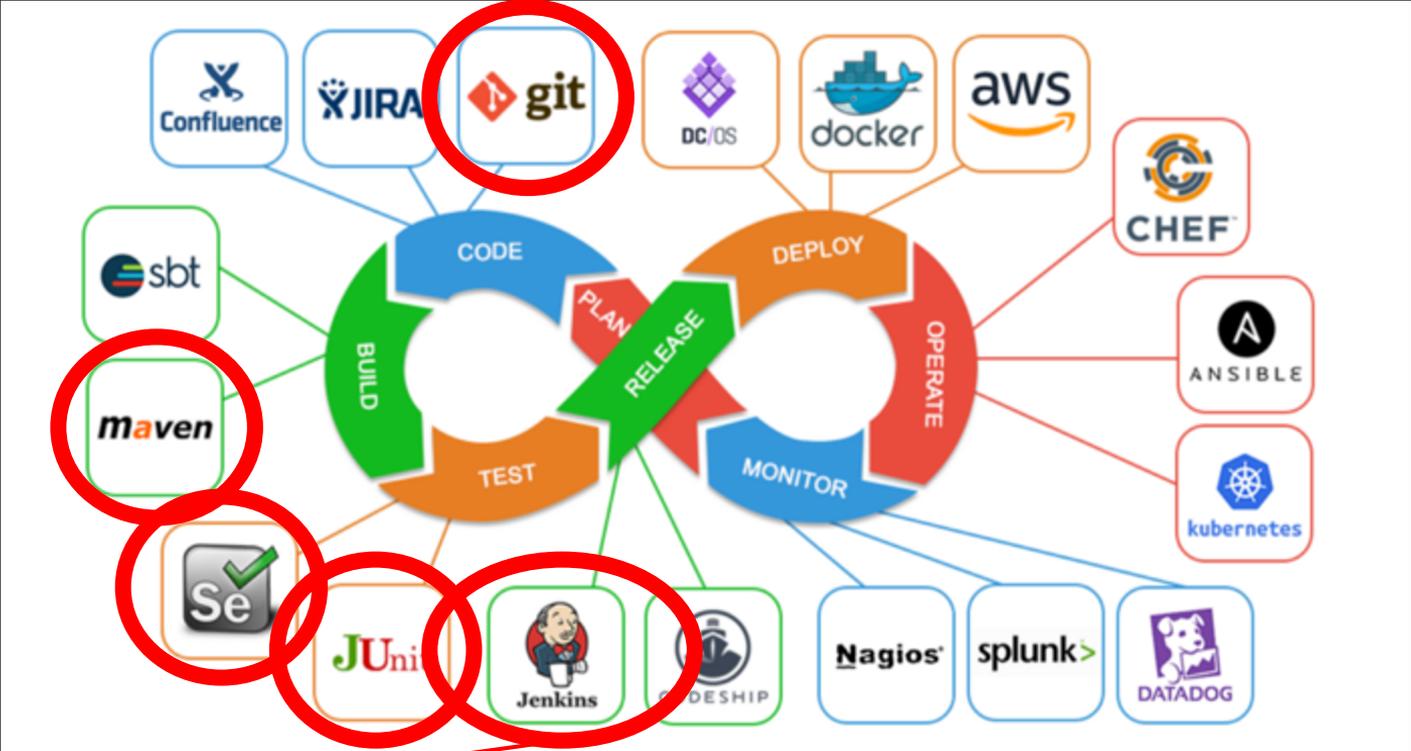


Regression:

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

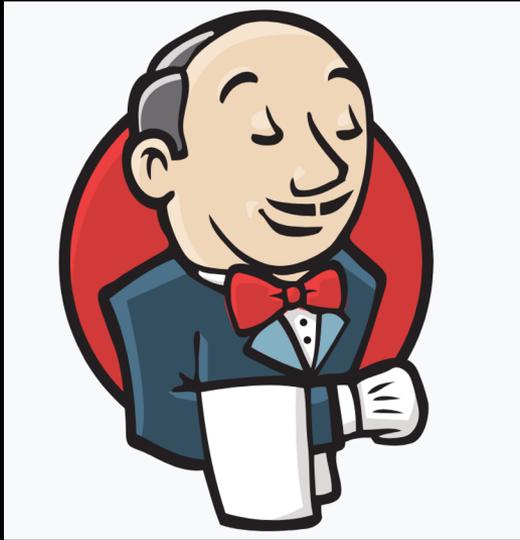


DevOps



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.

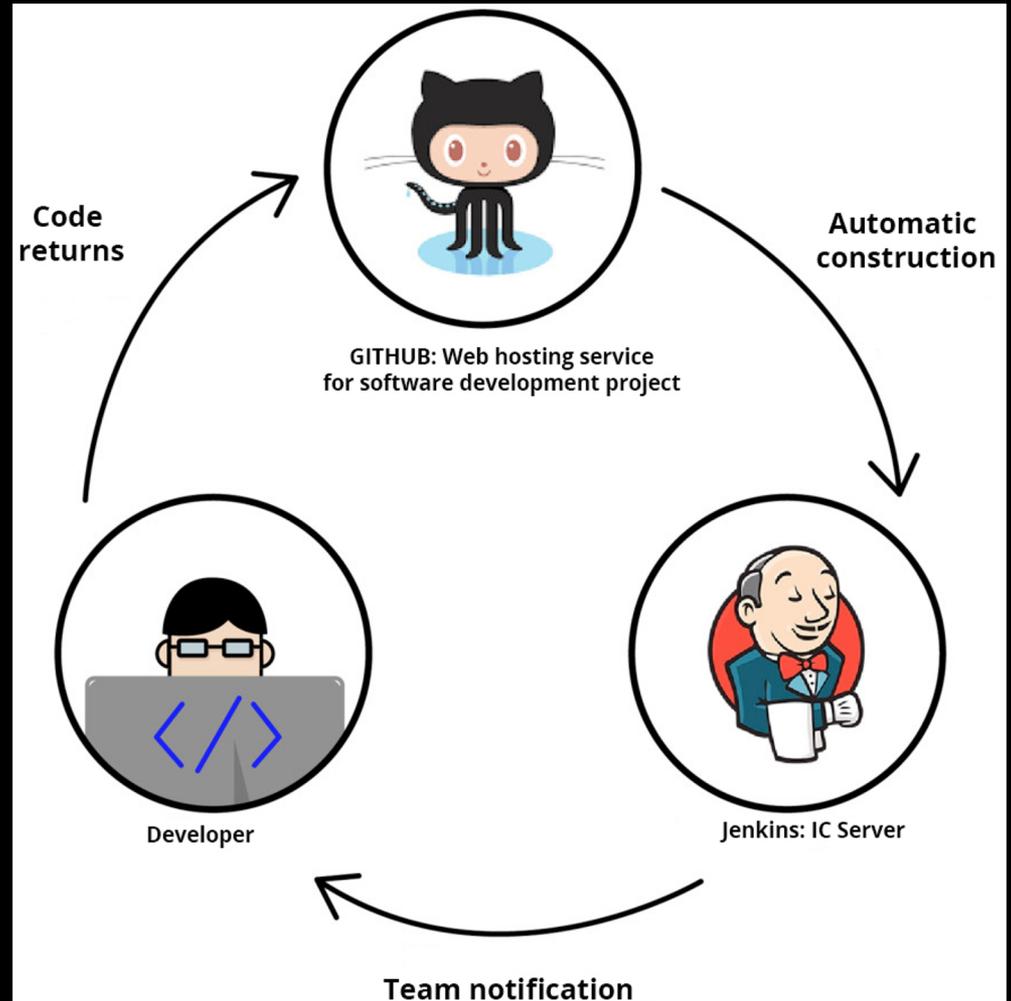
If something goes wrong, the tool immediately notifies the developers so that they can fix the issue immediately.

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!

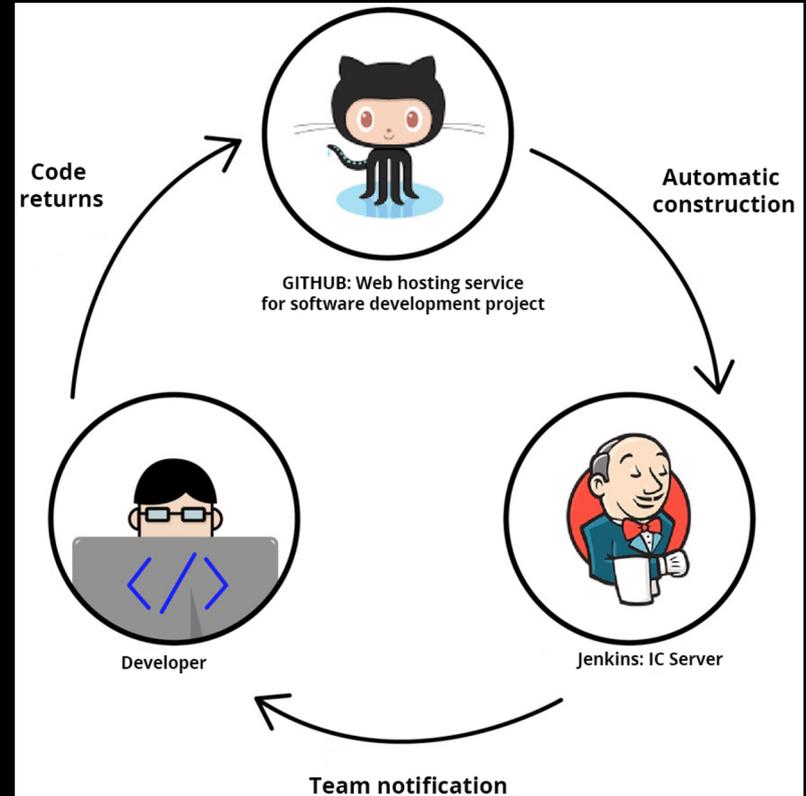
Continuous Integration with Jenkins

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/>

Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images.

1. Before downloading, please take a moment to review the [Hardware and Requirements](#) page.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the [Installing Jenkins](#) page.
4. You may also want to verify the package you downloaded. [Learn more about verifying packages](#).

Download Jenkins 2.249.3 LTS for:

Generic Java package (.war)

SHA-256: 8de8f11d5688c79967bc53a8124960926a90d623e5e9f03f1315ccf3e7c49702

Hands On

localhost:8080/manager/html

None specified	Welcome to Tomcat
None specified	Tomcat Document
None specified	Servlet and JSP E
None specified	Tomcat Host Mana
None specified	Tomcat Manager A



file located on server

Context Path:

Version (for parallel deployment):

XML Configuration file path:

WAR or Directory path:

Select WAR file to upload No file chosen

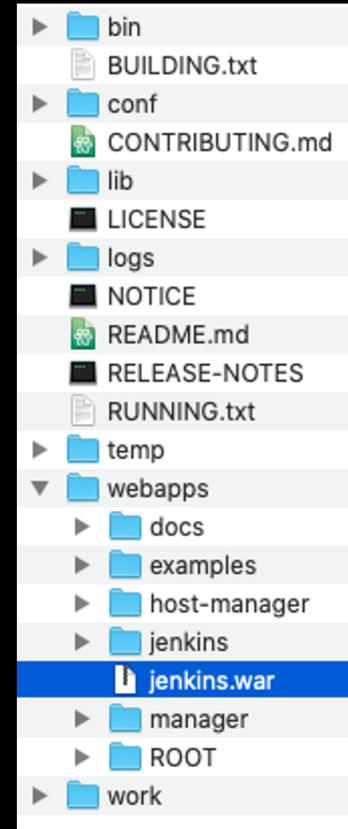
- Deploy Jenkins on Tomcat
- Choose File and select jenkins.war
- Click Deploy

WAR file to deploy

Select WAR file to upload jenkins.war

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



/jenkins	<i>None specified</i>	Jenkins v2.249.3	true	0	Start Stop Reload Undeploy
					Expire sessions with idle ≥ 30 minutes

Hands On

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

Attività iniziali

Sblocca Jenkins

Per assicurarsi che Jenkins venga configurato in modo sicuro dall'amministratore, è stata scritta una password nel log ([non si è sicuri di dove trovarla?](#)) e su questo file sul server:

```
C:\Users\studente\.jenkins\secrets\initialAdminPassword
```

Copiare la password da uno di tali percorsi e incollarla qui sotto.

Password amministratore

[Continua](#)

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

The screenshot shows the Jenkins installation progress page. The browser address bar indicates the URL is `localhost:8080/jenkins/`. The page title is "Attività iniziali". A progress bar is partially filled with blue. Below the progress bar, there is a table of installed and pending plugins. The installed plugins are marked with a green checkmark, and the pending ones with a blue refresh icon. On the right side, there is a list of plugins that are being installed or are dependencies.

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	** SSH server
✓ Timestampers	Workspace Cleanup	Ant	Gradle	Folders
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	OWASP Markup Formatter
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	** Structs
LDAP	Email Extension	Mailer		** Trilead API
				** Pipeline: Step API
				** Token Macro
				Build Timeout
				** JAXB
				** Credentials
				** Plain Credentials
				** SSH Credentials
				Credentials Binding
				** SCM API
				** Pipeline: API
				Timestampers
				** Caffeine API
				** Script Security
				** Plugin Utilities API
				** Font Awesome API
				** - dipendenza richiesta

Jenkins 2.321

Hands On

1. Create an admin user

Attività iniziali

Configurazione istanza

URL Jenkins:

L'URL di Jenkins è utilizzato per fornire l'URL radice per i collegamenti assoluti a varie risorse di Jenkins. Ciò significa che questo valore è richiesto per il corretto funzionamento di molte funzionalità di Jenkins incluse le notifiche inviate tramite posta elettronica, gli aggiornamenti di stato delle pull request e la variabile d'ambiente BUILD_URL fornita ai passaggi di compilazione.

L'impostazione predefinita proposta e visualizzata **non è ancora stata salvata** e, se possibile, è generata dalla richiesta corrente. La procedura consigliata è impostare questo valore all'URL che ci si aspetta che gli utenti utilizzino. Ciò eviterà che si generi confusione durante la condivisione o la visualizzazione di collegamenti.

Jenkins 2.249.3 Non ora [Salva e finisci](#)

<http://localhost:8080/jenkins/>

Attività iniziali

Crea primo utente amministratore

Nome utente:

Password:

Conferma password:

Nome completo:

Indirizzo di posta elettronica:

Jenkins 2.249.3 Continua come amministratore [Salva e continua](#)

Attività iniziali

Jenkins è pronto!

L'installazione di Jenkins è stata completata.

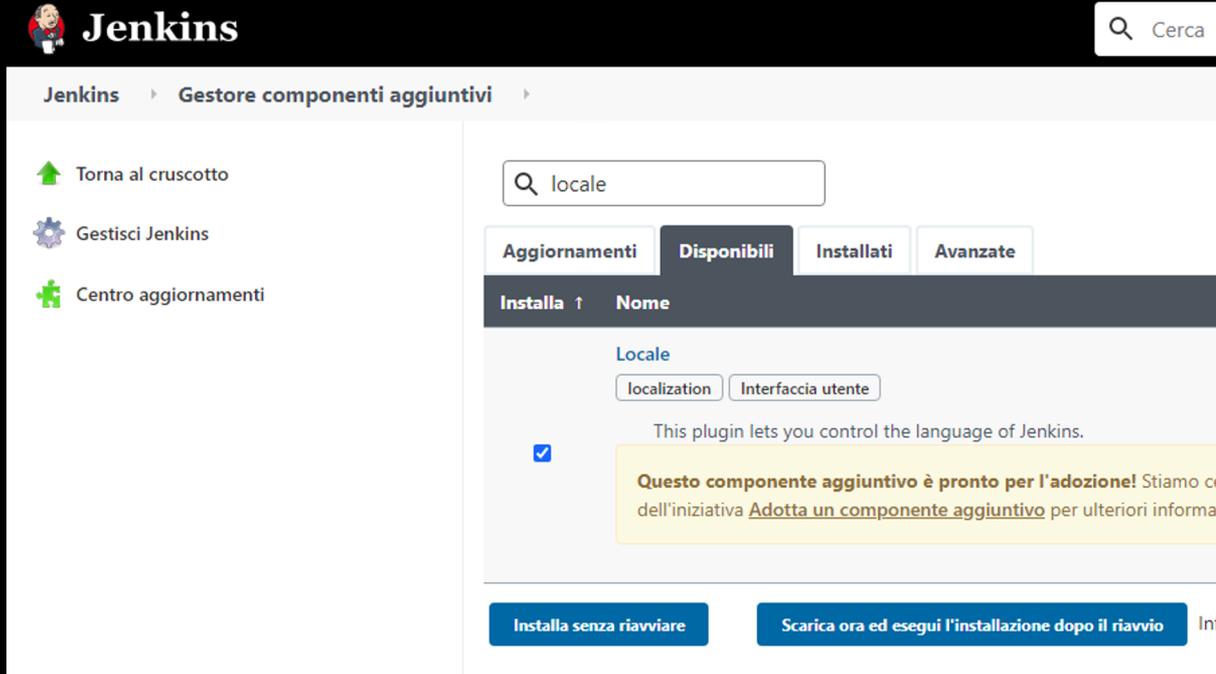
[Inizia ad utilizzare Jenkins](#)

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.



The screenshot shows the Jenkins 'Gestore componenti aggiuntivi' (Manage Plugins) page. The search filter is set to 'locale'. The 'Disponibili' (Available) tab is selected, showing the 'Locale' plugin. The plugin is checked for installation. Below the plugin name, there are two buttons: 'localization' and 'Interfaccia utente'. A yellow banner indicates that the plugin is ready for adoption. At the bottom, there are two buttons: 'Installa senza riavviare' (Install without restart) and 'Scarica ora ed esegui l'installazione dopo il riavvio' (Download now and run installation after restart).

Configure JDK

Manage Jenkins > Global Tool Configuration

JDK

JDK installations Add JDK

JDK

Name

JAVA_HOME

Install automatically ?

Delete JDK

Add JDK

List of JDK installations on this system

Configure JDK & Maven

Manage Jenkins --> Global Tool Configuration

Maven

Maven installations

Add Maven

Maven

Name

MAVEN_HOME

Install automatically

Delete Maven

Or

Tell Jenkins to
install a version

Maven

Maven installations

Add Maven

Maven

Name

Install automatically

Install from Apache

Version

Delete Installer

Create Job

1 Create a new Job

Enter an item name

» Required field

 **Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

2 Link it with your
GitHub Repository

Source Code Management

None
 Git

Repositories

Repository URL

Credentials

Run a Job

Run a Job

Jenkins > SPM2020 >

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now**
- Configure
- Delete Project
- Rename

Build History [trend](#) ^

find

Project SPM2020

This job has been created for the SPM2020 course

- Workspace
- Recent Changes

Permalinks

Run a Job

S	W	Name ↓	Last Success	Last Failure	Last Duration	
		SPM2020	2 min 7 sec - #5	4 min 36 sec - #3	2,4 sec	

Icon: S M L

Legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds