

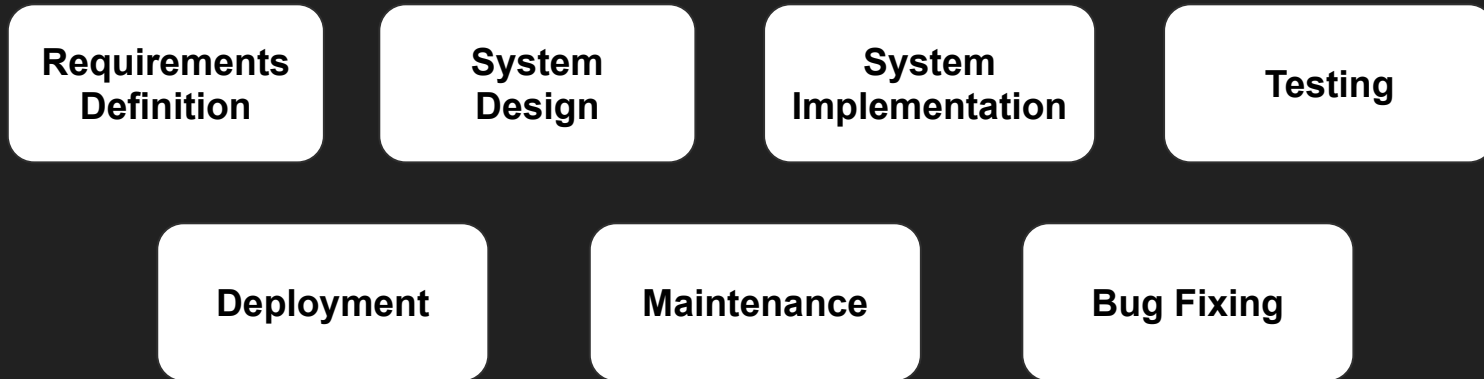
Software Project Management - Laboratory

Lecture n° 17
A.Y. 2021-2022

Prof. Fabrizio Fornari

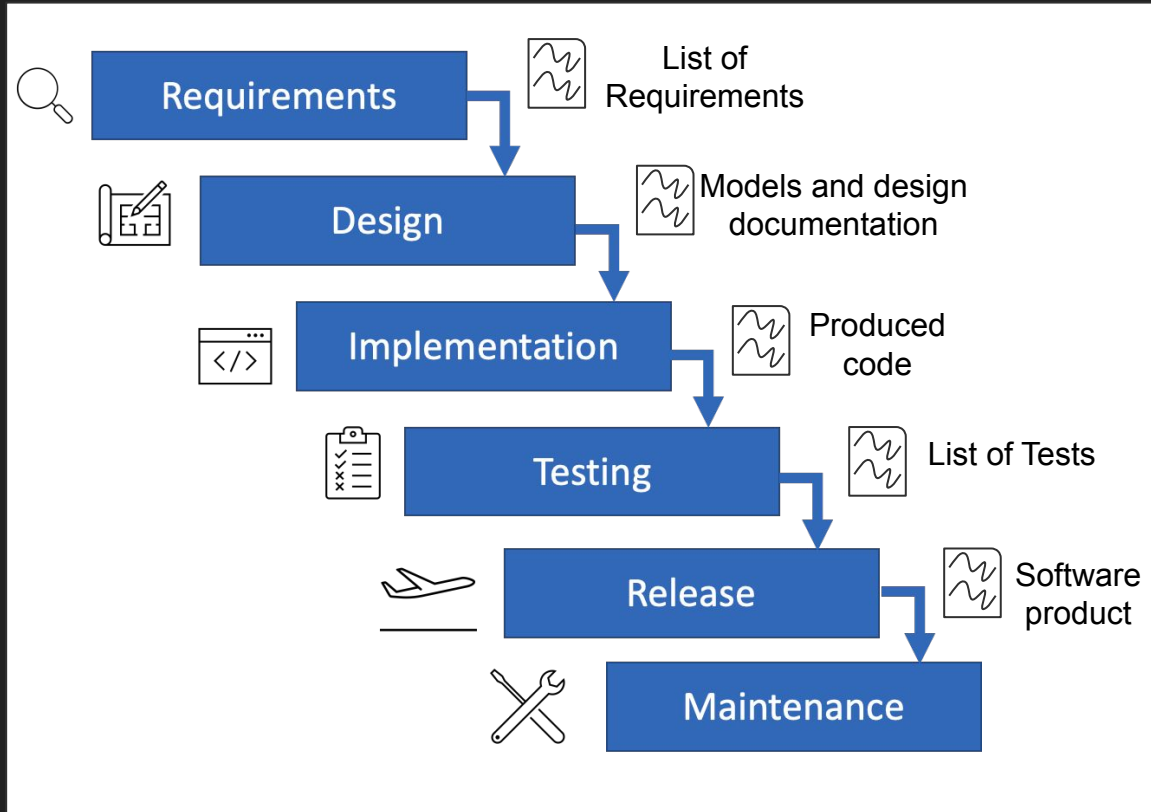
Software Development Process

Software Development Process is the process of dividing software development work into distinct phases to improve design, product management, and project management. It is also known as a software development life cycle (SDLC)

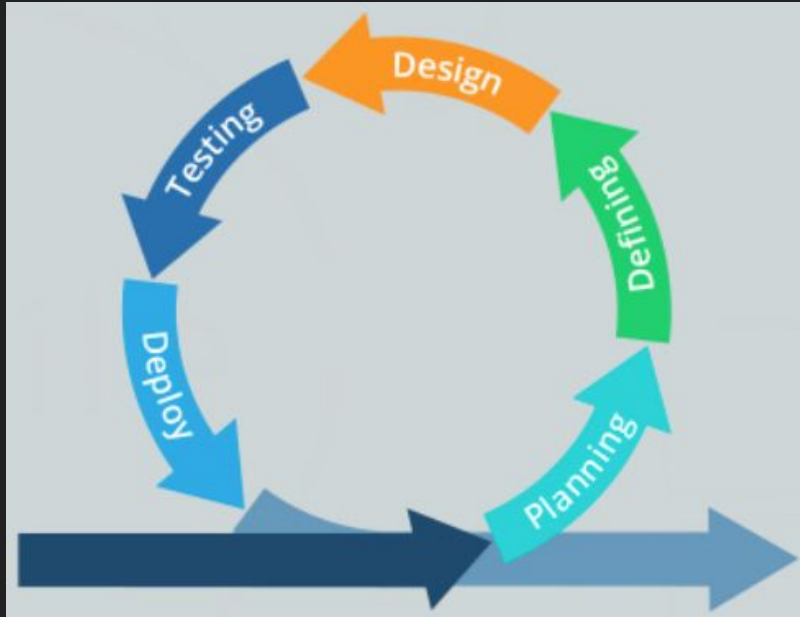


Waterfall Model - Negative Aspects

- Too much focused on the production of documents and less on the actual software product
- Software is released only at the end
- Customer involved only during the initial phase (requirements definition)
- Changings in the requirements are not possible after the requirements phase is over



Agile Development Process



Manifesto for Agile Software Development

Individuals and interactions over processes and tools

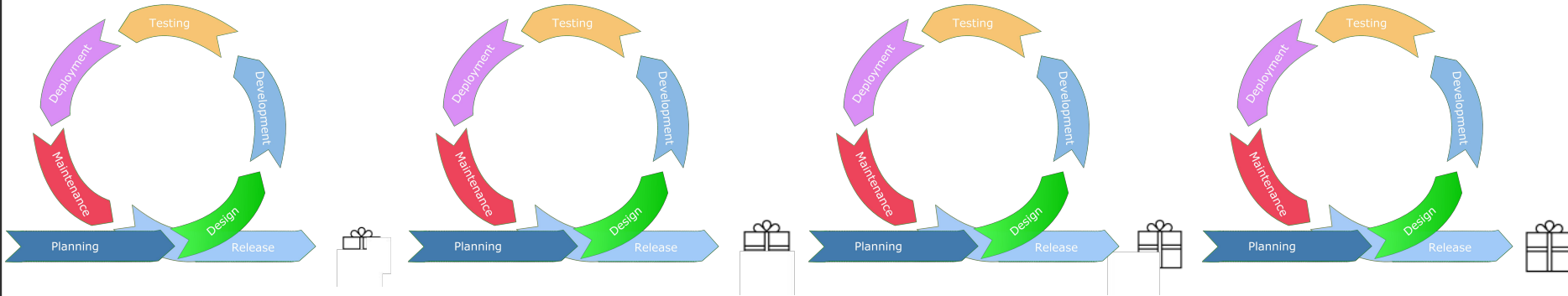
Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Waterfall vs Agile

The focus is on the software product



What about your project?

Does it have any kind of documentation?

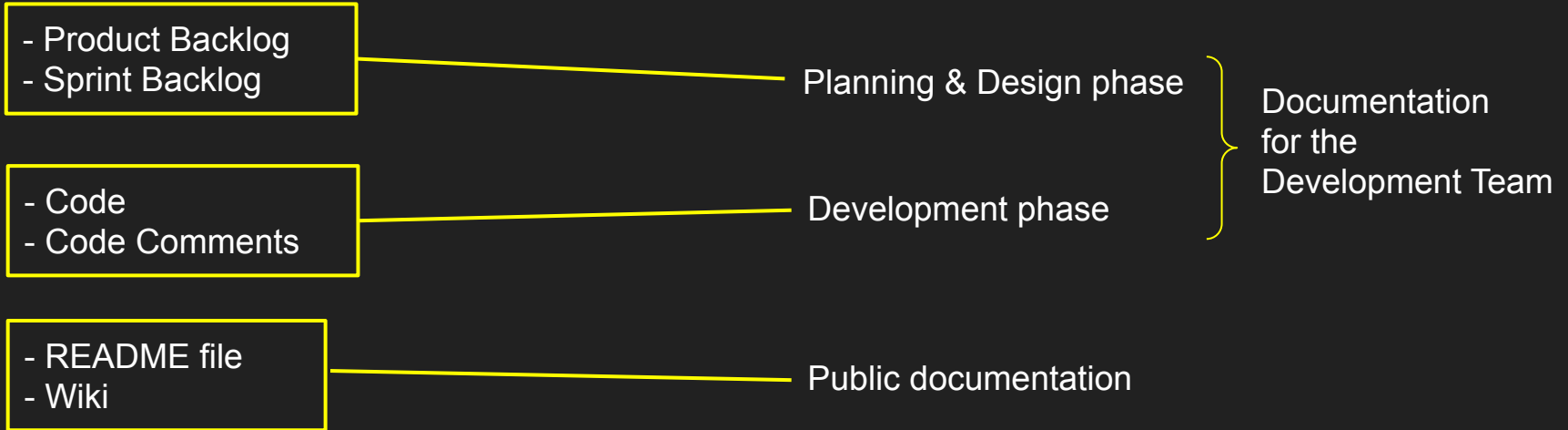
SCRUM Artifacts

The SCRUM artifacts are used to help define the workload coming into the team and currently being worked upon the team.

The main artifacts:

- Product backlog - a collection of user stories which present functionalities required/wanted by the product team. Usually the product owner takes responsible for this list.
- Sprint backlog - a collection of stories which could be included in the current sprint.

Which kind of documentation in a Agile/Scrum project?



README

You can add a README file to a repository to communicate important information about your project. A README, along with a repository license, contribution guidelines, and a code of conduct, communicates expectations for your project and helps you manage contributions

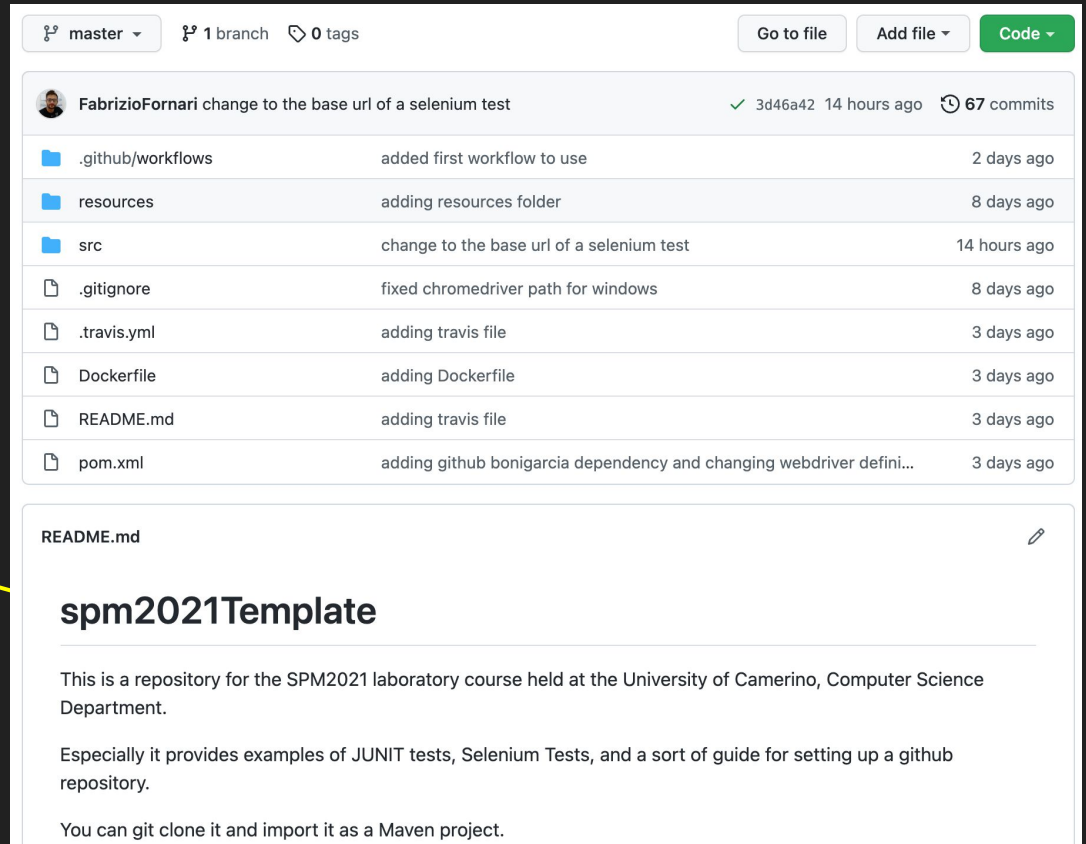
A README is often the first item a visitor will see when visiting your repository. README files typically include information on:

- **What the project does**
- **Why the project is useful**
- **How users can get started with the project**
- **Where users can get help with your project**
- **Who maintains and contributes to the project**

If you put your README file in your repository's root, `docs`, or hidden `.github` directory, GitHub will recognize and automatically surface your README to repository visitors.

README

README file



The screenshot shows a GitHub repository interface. At the top, it displays the current branch as 'master', 1 branch, and 0 tags. There are buttons for 'Go to file', 'Add file', and 'Code'. Below this, a commit by FabrizioFornari is shown, with a commit hash of 3d46a42 and 67 commits. The commit message is 'change to the base url of a selenium test'. A list of files and folders is shown, including .github/workflows, resources, src, .gitignore, .travis.yml, Dockerfile, README.md, and pom.xml. The README.md file is selected, and its content is displayed below. The content includes the title 'spm2021Template' and a description of the repository's purpose for the SPM2021 laboratory course at the University of Camerino, Computer Science Department. It also mentions that it provides examples of JUNIT tests, Selenium Tests, and a guide for setting up a github repository, and that it can be git cloned and imported as a Maven project.

master 1 branch 0 tags Go to file Add file Code

FabrizioFornari change to the base url of a selenium test ✓ 3d46a42 14 hours ago 67 commits

.github/workflows	added first workflow to use	2 days ago
resources	adding resources folder	8 days ago
src	change to the base url of a selenium test	14 hours ago
.gitignore	fixed chromedriver path for windows	8 days ago
.travis.yml	adding travis file	3 days ago
Dockerfile	adding Dockerfile	3 days ago
README.md	adding travis file	3 days ago
pom.xml	adding github bonigarcia dependency and changing webdriver defini...	3 days ago

README.md

spm2021Template

This is a repository for the SPM2021 laboratory course held at the University of Camerino, Computer Science Department.

Especially it provides examples of JUNIT tests, Selenium Tests, and a sort of guide for setting up a github repository.

You can git clone it and import it as a Maven project.

Github - Wiki

Every GitHub repository comes equipped with **a section for hosting documentation**, called a **wiki**. We can use our repository's wiki to share long-form content about our project, such as **how to use it**, **how we designed it**, or **its core principles**. We can use a wiki to provide additional documentation.

If you create a wiki in a public repository, the wiki is available to the public. If you create a wiki in an internal or private repository, people with access to the repository can also access the wiki.

You can edit wikis directly on GitHub, or you can edit wiki files locally. By default, only people with write access to your repository can make changes to wikis, although you can allow everyone on GitHub to contribute to a wiki in a public repository.

Cloning wikis to your computer

```
$ git clone https://github.com/YOUR_USERNAME/YOUR_REPOSITORY.wiki.git  
# Clones the wiki locally
```

Github - Wiki








FabrizioFornari / SPM2020Template Unwatch 1

<> Code Issues 4 Pull requests Actions Projects 3 **Wiki** Security Insights Settings

Create new page

Home

Write Preview

h1 h2 h3   **B** *i* <>      Edit mode: Markdown

Welcome to the SPM2020Template wiki!

Edit message

Write a small message here explaining this change. (Optional)

Save Page

Github - Wiki Permissions

Features

- Wikis
- Restrict editing to collaborators only**
Public wikis will still be readable by everyone.

Wiki spm2021Template

FabrizioFornari / **spm2021Template** Public

Unwatch 1 Star 0 Fork 18

Code Issues Pull requests Actions Projects **Wiki** Security Insights Settings

Home

Fabrizio Fornari edited this page 3 minutes ago · 1 revision

Welcome to the spm2021Template wiki!

This project consists of a Maven Project that reports some examples of the usage of Maven, JUnit, Selenium, Jenkins, Tomcat etc. This material is meant to be used by the students of the SPM2021 course.

In addition the GitHub repository has been configured to apply Scrum. Especially the "Issues" section is used as the Product Backlog, where user stories are reported, and Sprint Backlogs (in the form of Milestone) are defined. The "Projects" section is used to divide user stories into tasks and to keep track of their progress in the form of a Kanban board.

SPM2021 Course University of Camerino Prof. Fabrizio Fornari

Pages 8

Wiki

- [Home](#)

Content

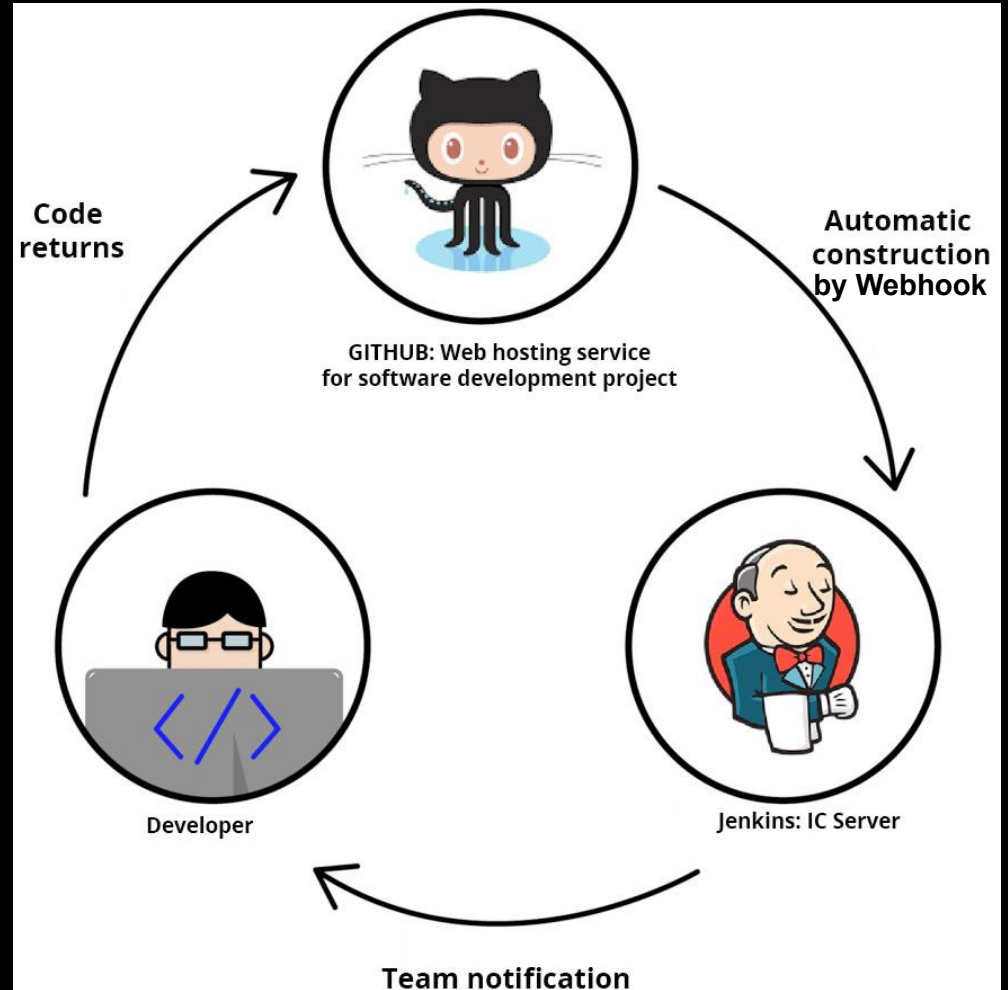
- [Scrum](#)
- [DevOps](#)
- [Maven](#)
- [JUnit](#)
- [Selenium](#)
- [Jenkins](#)
- [Tomcat](#)

Clone this wiki locally

<https://github.com/FabrizioFornari>

Continuous Integration with Jenkins

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



Build Status

README.md



SPM2020Template

This is a repository for the SPM2020 laboratory course held at the University of Camerino, Computer Science Department.

Especially it provides examples of JUNIT tests, Selenium Tests, and a sort of guide for setting up a github repository.

You can git clone it and import it as a Maven project.

license [GPL \(>= 2\)](#)

<https://shields.io>

[Jenkins] [build passing](#)

[Travis CI] [build canceled](#)

Build Status



Jenkins

search



Fabrizio Fornari

log out

Jenkins > Plugin Manager >

Back to Dashboard

Manage Jenkins

Embeddable Build Status

Updates

Available

Installed

Advanced

Install ↑

Name

Version

Released

[Embeddable Build Status](#)



User Interface

2.0.3

1 yr 1 mo ago

This plugin adds the embeddable build status badge to Jenkins so that you can easily hyperlink/show your build status from elsewhere.

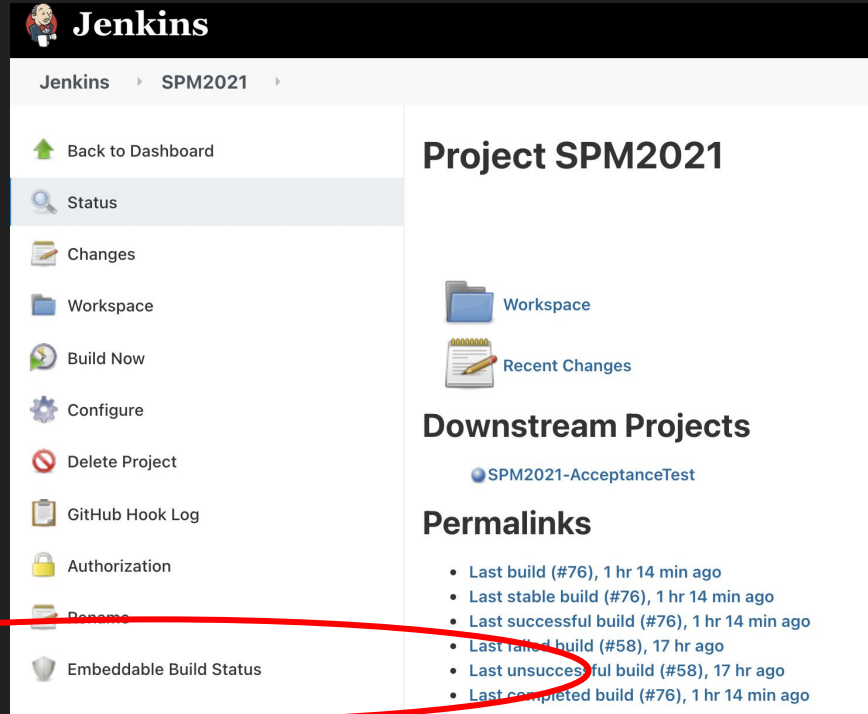
Install without restart

Download now and install after restart

Update information obtained: 1 day 11 hr ago

Check now

Embeddable Build Status



The screenshot shows the Jenkins interface for Project SPM2021. The left sidebar contains a menu with the following items: Back to Dashboard, Status (highlighted), Changes, Workspace, Build Now, Configure, Delete Project, GitHub Hook Log, Authorization, Rename, and Embeddable Build Status. The main content area displays the Project SPM2021 header, followed by Workspace and Recent Changes sections. Below these are Downstream Projects (SPM2021-AcceptanceTest) and a Permalinks section with a list of build status links. A red oval highlights the 'Embeddable Build Status' option in the sidebar and the 'Last failed build (#58), 17 hr ago' link in the Permalinks section.

Jenkins

Jenkins > SPM2021

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Project

GitHub Hook Log

Authorization

Rename

Embeddable Build Status

Project SPM2021

Workspace

Recent Changes

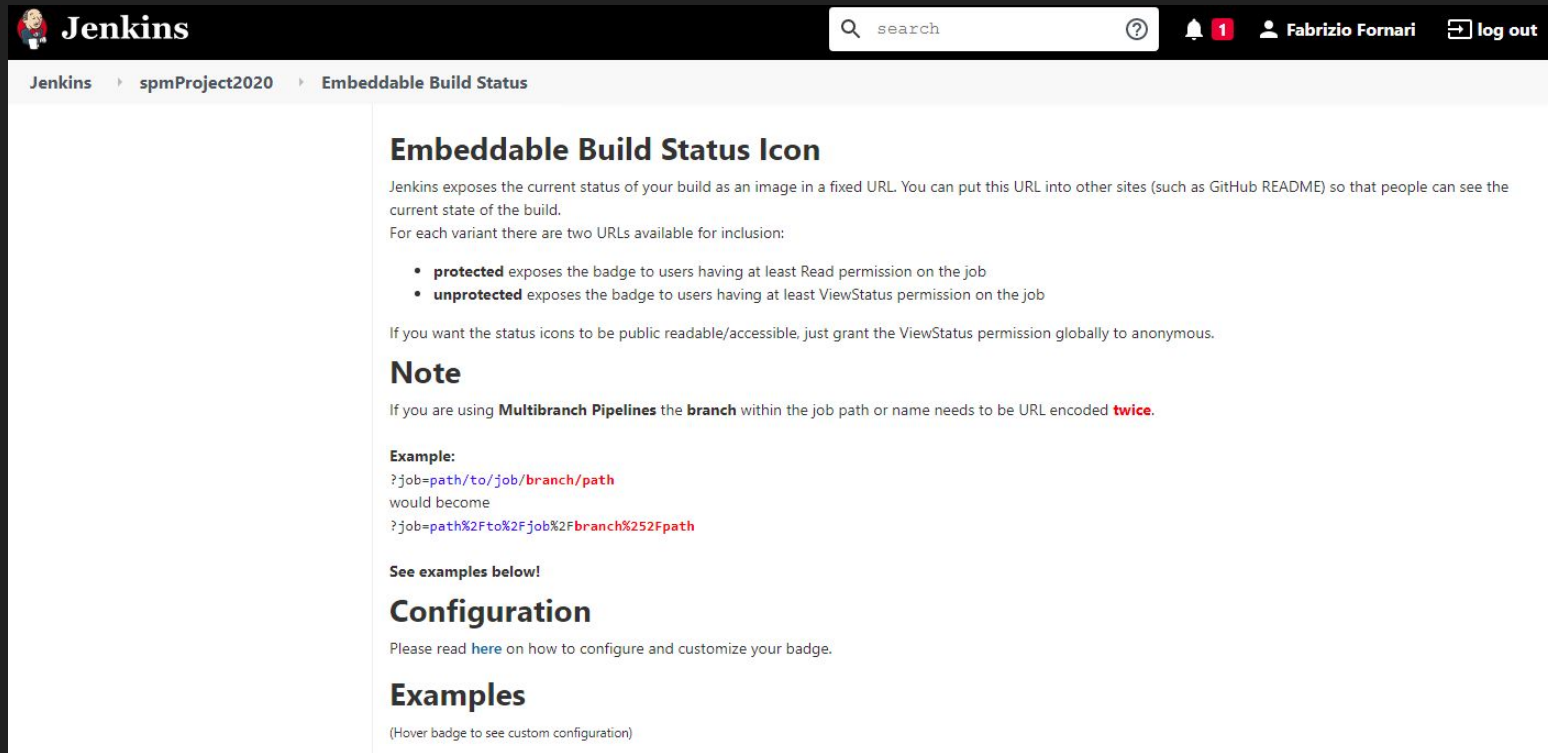
Downstream Projects

SPM2021-AcceptanceTest

Permalinks

- Last build (#76), 1 hr 14 min ago
- Last stable build (#76), 1 hr 14 min ago
- Last successful build (#76), 1 hr 14 min ago
- Last failed build (#58), 17 hr ago
- Last unsuccessful build (#58), 17 hr ago
- Last completed build (#76), 1 hr 14 min ago

Embeddable Build Status



The screenshot shows the Jenkins web interface. At the top, there's a navigation bar with the Jenkins logo, a search bar, a notification bell with a '1', the user name 'Fabrizio Fornari', and a 'log out' button. Below the navigation bar, the breadcrumb trail reads 'Jenkins > spmProject2020 > Embeddable Build Status'. The main content area has a heading 'Embeddable Build Status Icon'. The text explains that Jenkins exposes the current status of a build as an image in a fixed URL, which can be used in other sites like GitHub README. It lists two URL variants: 'protected' (for users with Read permission) and 'unprotected' (for users with ViewStatus permission). A note mentions that for public readability, the ViewStatus permission should be granted globally to anonymous users. An example shows how a path with slashes is URL-encoded. A 'Configuration' section points to a link for more details. Finally, an 'Examples' section is partially visible with a note to hover over a badge for configuration.

Jenkins

search

1 Fabrizio Fornari log out

Jenkins > spmProject2020 > Embeddable Build Status

Embeddable Build Status Icon

Jenkins exposes the current status of your build as an image in a fixed URL. You can put this URL into other sites (such as GitHub README) so that people can see the current state of the build.

For each variant there are two URLs available for inclusion:

- **protected** exposes the badge to users having at least Read permission on the job
- **unprotected** exposes the badge to users having at least ViewStatus permission on the job

If you want the status icons to be public readable/accessible, just grant the ViewStatus permission globally to anonymous.

Note

If you are using **Multibranch Pipelines** the **branch** within the job path or name needs to be URL encoded **twice**.

Example:

```
?job=path/to/job/branch/path
```

would become

```
?job=path%2Fto%2Fjob%2Fbranch%252Fpath
```

See examples below!

Configuration

Please read [here](#) on how to configure and customize your badge.

Examples

(Hover badge to see custom configuration)

Markdown:

```
[[Build Status](http://proslabtest.unicam.it/jenkins/buildStatus/icon?job=SPM2021)](http://proslabtest.unicam.it/jenkins/job/SPM2021/)
```

Build Status Markdown on GitHub



```
spm2021Template / README.md in master  
<> Edit file Preview Spaces 2 Soft  
1 # spm2021Template  
2  
3 This is a repository for the SPM2021 laboratory course held at the University of Camerino, Computer Science Department.  
4  
5 Especially it provides examples of JUNIT tests, Selenium Tests, and a sort of guide for setting up a github repository.  
6  
7 You can git clone it and import it as a Maven project.  
8  
9 ![CRAN/METACRAN](https://img.shields.io/cran/l/devtools.svg)  
10  
11 https://shields.io  
12  
13 [Jenkins]  
14 [!Build Status](http://proslabtest.unicam.it/jenkins/buildStatus/icon?job=SPM2021)(http://proslabtest.unicam.it/jenkins/job/SPM2021/)
```

Build Status Configure Job

Set GitHub commit status (universal) X ?

Where:

Commit SHA: Latest build revision ?

Repositories: Any defined in job repository ?

What:

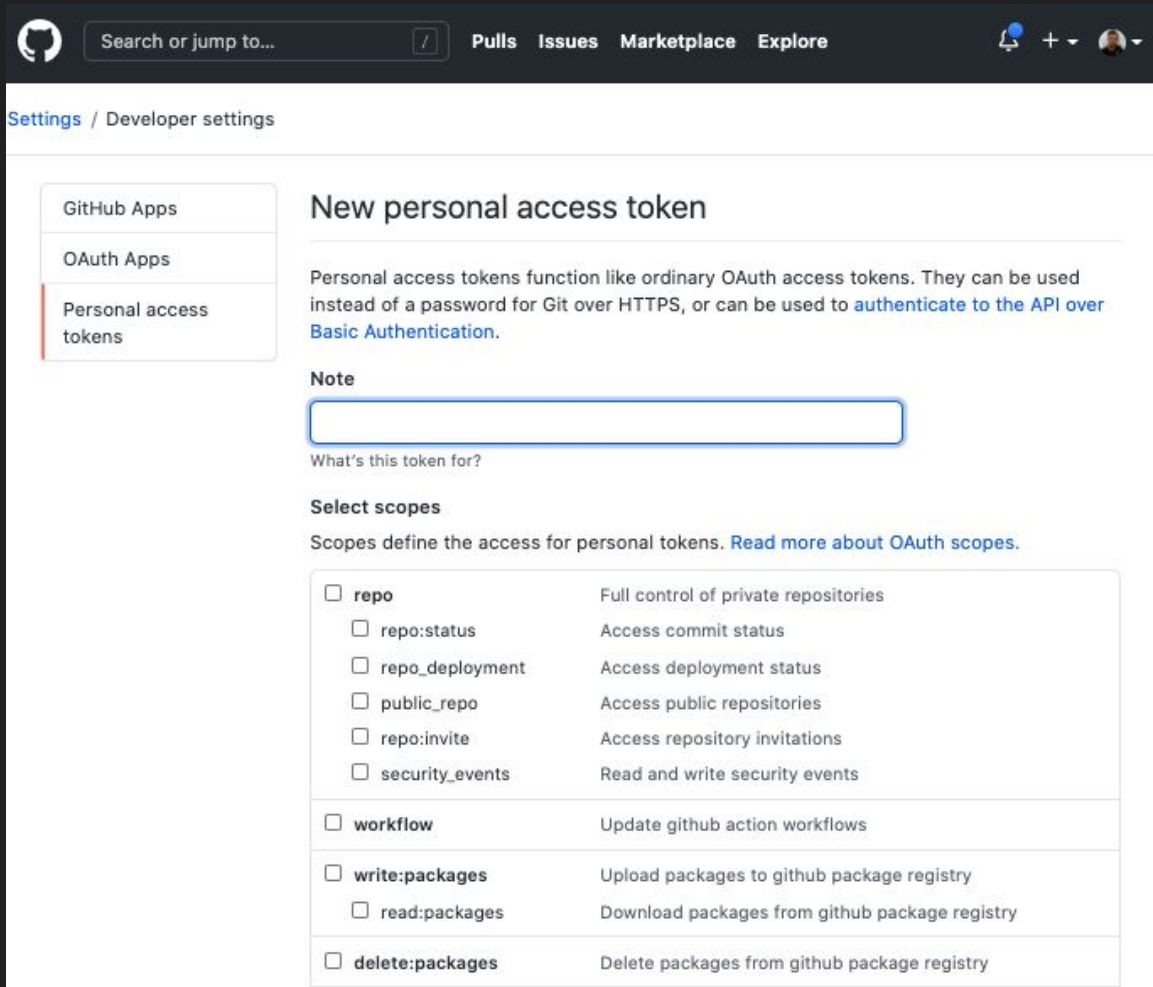
Commit context: From GitHub property with fallback to job name ?

Status result: One of default messages and statuses ?

Status backref: Backref to the build ?

[Advanced...](#)

GitHub Secret Text



The screenshot shows the GitHub interface for creating a new personal access token. At the top, there is a navigation bar with the GitHub logo, a search bar, and links for Pulls, Issues, Marketplace, and Explore. Below this, the breadcrumb 'Settings / Developer settings' is visible. On the left, a sidebar menu contains 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens', with the latter being selected. The main content area is titled 'New personal access token' and includes an introductory paragraph, a 'Note' section with an empty text box, a 'Select scopes' section with a list of checkboxes and their descriptions, and a 'What's this token for?' label.

Search or jump to... / Pulls Issues Marketplace Explore

Settings / Developer settings

- GitHub Apps
- OAuth Apps
- Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

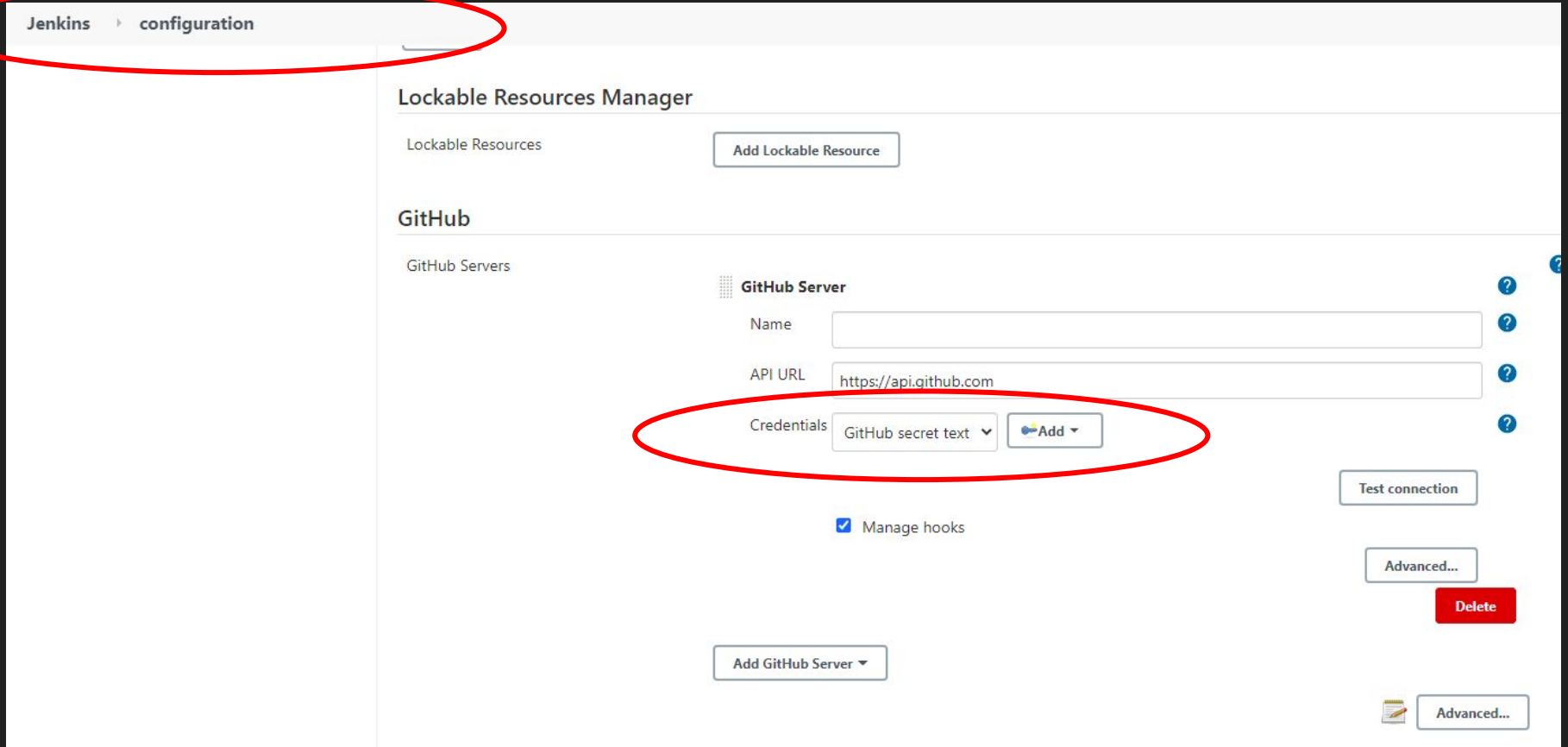
What's this token for?

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo:invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update github action workflows
<input type="checkbox"/> write:packages	Upload packages to github package registry
<input type="checkbox"/> read:packages	Download packages from github package registry
<input type="checkbox"/> delete:packages	Delete packages from github package registry

GitHub Secret On Jenkins Credentials



The image shows the Jenkins configuration page for a GitHub server. The breadcrumb navigation at the top left is "Jenkins > configuration", which is circled in red. Below this, the "Lockable Resources Manager" section is visible. The main section is titled "GitHub" and contains "GitHub Servers". A single server is listed with the following fields: "Name" (empty), "API URL" (https://api.github.com), and "Credentials" (GitHub secret text). The "Credentials" field and its "Add" button are circled in red. Below the "Credentials" field is a checked checkbox for "Manage hooks". At the bottom right, there are buttons for "Test connection", "Advanced...", "Delete", and another "Advanced..." button.

Jenkins > configuration

Lockable Resources Manager

Lockable Resources [Add Lockable Resource](#)

GitHub

GitHub Servers

GitHub Server

Name

API URL

Credentials [Add](#)

Manage hooks

[Test connection](#)

[Advanced...](#)

[Delete](#)

[Add GitHub Server](#)

[Advanced...](#)

Build Status

README.md



SPM2020Template

This is a repository for the SPM2020 laboratory course held at the University of Camerino, Computer Science Department.

Especially it provides examples of JUNIT tests, Selenium Tests, and a sort of guide for setting up a github repository.

You can git clone it and import it as a Maven project.

license [GPL \(>= 2\)](#)

<https://shields.io>

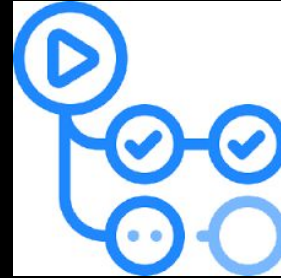
[Jenkins] [build](#) [passing](#)

Not only Jenkins



TRAVIS CI

<https://travis-ci.com/>



GitHub Actions

<https://docs.github.com/en/actions>

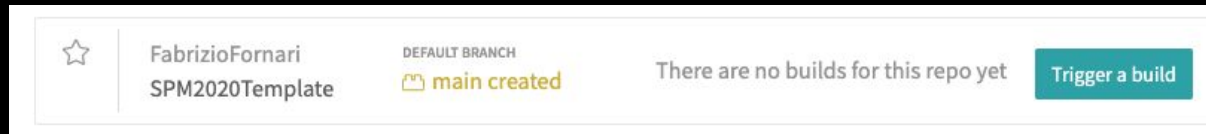
Travis CI

Travis CI is a hosted, distributed continuous integration service used to build and test software projects hosted at GitHub.

Travis CI is configured by adding a file named `.travis.yml`, which is a YAML format text file, to the root directory of the repository.

```
# this is a java project using maven
language: java
# install
install: mvn install
```


<https://travis-ci.com/>



The screenshot shows a Travis CI repository page for 'FabrizioFornari SPM2020Template'. It features a star icon, the repository name, the default branch 'main created', and a message stating 'There are no builds for this repo yet' with a 'Trigger a build' button.




Travis CI Documentation: <https://docs.travis-ci.com/>


Travis CI

Travis CI  [Dashboard](#) [Changelog](#) [Documentation](#) [Help](#)

Search all repositories

[My Repositories](#) +


-  FabrizioFornari/BasicJUnitTests # 24
Duration: -
-  **FabrizioFornari/SPM2020Temple # 12**
Duration: 56 sec
Finished: about a minute ago
-  ISTI-FMT-LearnPAD/bpmnVerify # 12
Duration: 25 sec
Finished: 5 years ago


FabrizioFornari / SPM2020Template  **build passing**


[Current](#) [Branches](#) [Build History](#) [Pull Requests](#) > [Build #11](#)


✓ **main** removing a white line from travis file ↻ #11 passed


- ↻ Commit 0b33c5a [↗](#)
- ↻ Compare f73b5a7..0b33c5a [↗](#)
- ↻ Branch main [↗](#)

 Fabrizio Fornari

 </> Java

 AMD64

 Ran for 1 min 4 sec

 about a minute ago

Commit Status

FabrizioFornari / SPM2020Template

<> Code ! Issues **4** 🔗 Pull requests ▶ Actions 📁 Projects **3** 📖 Wiki 🛡️ Security 📈 Insights ⚙️ Settings

🔗 main ▾ 🔗 1 branch 🏷️ 0 tags

Go to file

Add file ▾

📄 Code ▾



FabrizioFornari Update README.md

● 1a88ba3 11 minutes ago 🕒 124 commits

📁 .settings

📁 WebContent

📁 drivers

📁 resources

Some checks haven't completed yet

1 pending and 1 successful checks



continuous-integrati...

[Details](#)



spmProject2020 — ...

[Details](#)

17 days ago

17 days ago

21 days ago

webapp

18 days ago

GitHub Actions

Get executed on GitHub Server

Jobs are execute on virtual machines hosted by GitHub.

The screenshot shows the GitHub Actions interface for the repository 'FabrizioFornari / spm2021Template'. The page is titled 'Choose a workflow template' and provides instructions on how to build, test, and deploy code. It lists several workflow templates categorized into 'Workflows made for your repository' and 'Deploy your code with these popular services'.

Choose a workflow template
Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow template to get started.
Skip this and [set up a workflow yourself](#) →

Workflows made for your repository Suggested

- Publish Java Package with Maven**
By GitHub Actions
Build a Java Package using Maven and publish to GitHub Packages.
[Set up this workflow](#)

```
mvn -B package --file pom.xml  
mvn deploy -s $GITHUB_WORKSPACE/settings.xml
```


actions/starter-workflows Java
- Java with Maven**
By GitHub Actions
Build and test a Java project with Apache Maven.
[Set up this workflow](#)

```
mvn -B package --file pom.xml
```


actions/starter-workflows Java
- Android CI**
By GitHub Actions
Build an Android project with Gradle.
[Set up this workflow](#)

```
chmod +x gradlew  
./gradlew build
```


actions/starter-workflows Java
- Java with Ant**
By GitHub Actions
Build and test a Java project with Apache Ant.
[Set up this workflow](#)

```
ant -noinput -buildfile build.xml
```


actions/starter-workflows Java

Deploy your code with these popular services

- Deploy Node.js to Azure Web App**
By Microsoft Azure
Build a Node.js project and deploy it to an Azure Web App.
[Set up this workflow](#)
actions/starter-workflows Deployment
- Deploy to Alibaba Cloud ACK**
By Alibaba Cloud
Deploy a container to Alibaba Cloud Container Service for Kubernetes (ACK).
[Set up this workflow](#)
actions/starter-workflows Deployment
- Deploy to Amazon ECS**
By Amazon Web Services
Deploy a container to an Amazon ECS service powered by AWS Fargate or Amazon EC2.
[Set up this workflow](#)
actions/starter-workflows Deployment

GitHub Actions

```
name: Java CI with Maven
```

```
on:
```

```
  push:
```

```
    branches: [ master ]
```

```
  pull_request:
```

```
    branches: [ master ]
```

```
jobs:
```

```
  build:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

```
      - uses: actions/checkout@v2
```

```
      - uses: browser-actions/setup-chrome@latest
```

```
      - name: Set up JDK 11
```

```
        uses: actions/setup-java@v2
```

```
        with:
```

```
          java-version: '11'
```

```
          distribution: 'adopt'
```

```
          cache: maven
```

```
      - name: Build with Maven
```

```
        run: mvn -B package --file pom.xml test
```

GitHub Actions

```
name: Java CI with Maven
```

The name of the workflow as it will appear in the Actions tab of the GitHub repository.

```
on:
```

```
  push:
    branches: [ master ]
  pull_request:
    branches: [ master ]
```

The trigger for this workflow. This example uses the push event, so a workflow run is triggered every time someone pushes a change to the repository or merges a pull request

```
jobs:
```

```
  build:
    runs-on: ubuntu-latest
```

Groups together all the jobs that run in the

```
  steps:
```

```
    - uses: actions/checkout@v2
    - uses: browser-actions/setup-chrome@latest
    - name: Set up JDK 11
      uses: actions/setup-java@v2
      with:
        java-version: '11'
        distribution: 'adopt'
        cache: maven
    - name: Build with Maven
      run: mvn -B package --file pom.xml test
```

Defines a job named build. The child keys will define properties of the job.

Configures the job to run on the latest version of an Ubuntu Linux runner

GitHub Actions

```
name: Java CI with Maven
```

```
on:  
  push:  
    branches: [ master ]  
  pull_request:  
    branches: [ master ]
```

```
jobs:  
  build:  
  
    runs-on: ubuntu-latest
```

```
  steps:  
    - uses: actions/checkout@v2  
    - uses: browser-actions/setup-chrome@latest  
    - name: Set up JDK 11  
      uses: actions/setup-java@v2  
      with:  
        java-version: '11'  
        distribution: 'adopt'  
        cache: maven  
    - name: Build with Maven  
      run: mvn -B package --file pom.xml test
```

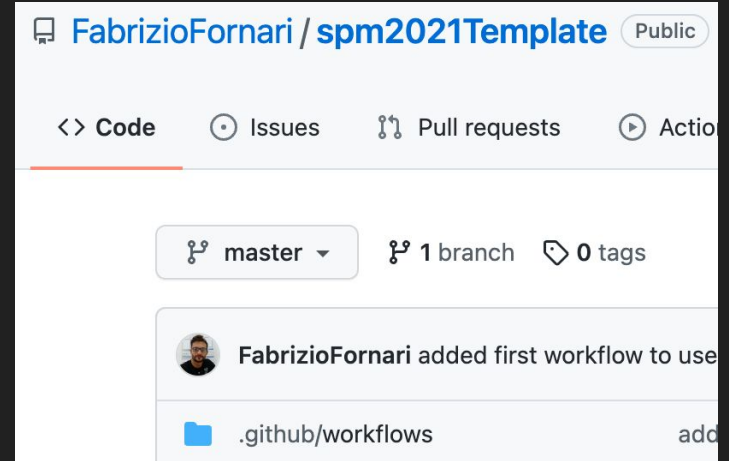
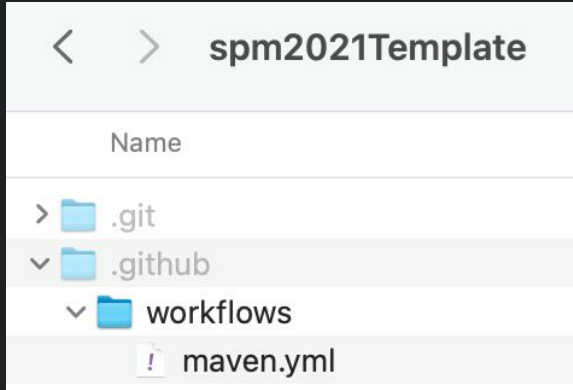
Groups together all the steps that run in the build job

The uses keyword specifies that this step will run v2 of the actions/checkout action (<https://github.com/actions>)

actions/setup-java@v2 action to install the specified version of java '11'

The run keyword tells the job to execute a command on the runner. In this case, we are using mvn to build our application and to run tests

.github/workflows



GitHub Actions

```
name: Java CI with Maven
```

```
on:  
  push:  
    branches: [ master ]  
  pull_request:  
    branches: [ master ]
```

```
jobs:  
  build:  
  
    runs-on: ubuntu-latest
```

```
  steps:  
    - uses: actions/checkout@v2  
    - uses: browser-actions/setup-chrome@latest  
    - name: Set up JDK 11  
      uses: actions/setup-java@v2  
      with:  
        java-version: '11'  
        distribution: 'adopt'  
        cache: maven  
    - name: Build with Maven  
      run: mvn -B package --file pom.xml test
```

The screenshot shows a GitHub Actions workflow run page. At the top, the repository is identified as 'FabrizioFornari / spm2021Template' (Public). The workflow name is 'changed selenium test for githubaction Java CI with Maven #16'. The page includes navigation tabs for Code, Issues, Pull requests, Actions (selected), Projects, Wiki, Security, Insights, and Settings. A 'Summary' section shows the workflow file path: '.github/workflows/maven.yml at 861c711'. The 'Jobs' section lists a single job named 'build' with a green checkmark. The 'Workflow file for this run' section displays the following YAML code:

```
1 # This workflow will build a Java project with Maven, and cache/restore any dependencies to improve the workflow execution  
2 # For more information see: https://help.github.com/actions/language-and-framework-guides/building-and-testing-java-with-maven  
3  
4 name: Java CI with Maven  
5  
6 on:  
7   push:  
8     branches: [ master ]  
9   pull_request:  
10    branches: [ master ]  
11  
12 jobs:  
13   build:  
14  
15     runs-on: ubuntu-latest  
16  
17     steps:  
18     - uses: actions/checkout@v2  
19     - uses: browser-actions/setup-chrome@latest  
20     - name: Set up JDK 11  
21       uses: actions/setup-java@v2  
22       with:  
23         java-version: '11'  
24         distribution: 'adopt'  
25         cache: maven  
26     - name: Install Google Chrome  
27       run: chrome --version  
28     - name: Build with Maven  
29       run: mvn -B package --file pom.xml test
```

Acceptance Tests

What happens with Selenium

WebDriverManager

WebDriverManager is an open-source Java library that carries out the management (i.e., download, setup, and maintenance) of the drivers required by Selenium WebDriver (e.g., chromedriver, geckodriver, msedgedriver, etc.) in a fully automated manner.

<https://github.com/bonigarcia/webdrivermanager>

Add dependency to pom.xml

```
<dependency>
  <groupId>io.github.bonigarcia</groupId>
  <artifactId>webdrivermanager</artifactId>
  <version>4.3.1</version>
</dependency>
```

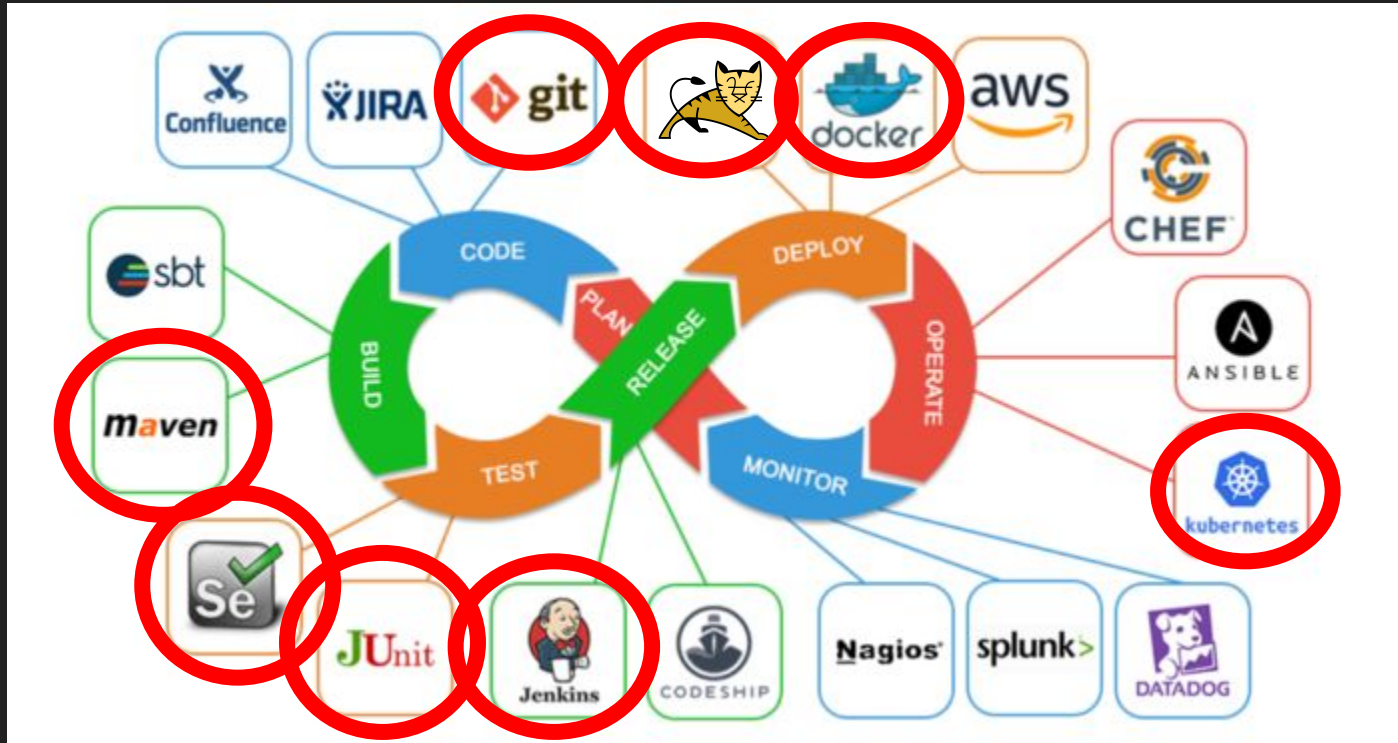
Adjust a selenium test

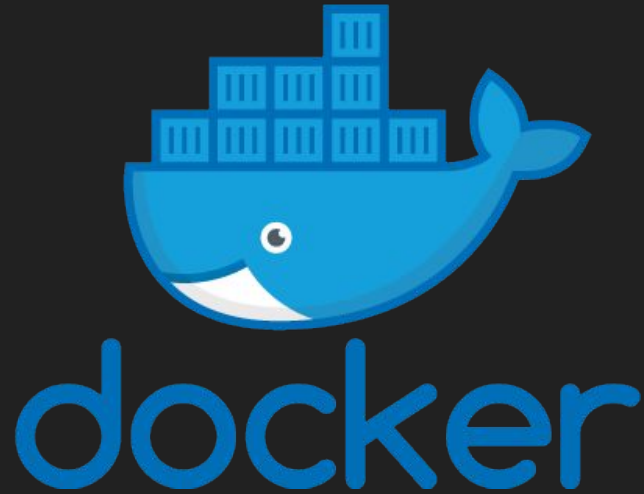
```
WebDriverManager.chromedriver().setup();
ChromeOptions options = new ChromeOptions()
options.addArguments("--no-sandbox");
options.addArguments("--disable-dev-shm-usage");
options.addArguments("--headless");
driver = new ChromeDriver(options);
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
```

Looking ahead...



DevOps Technologies





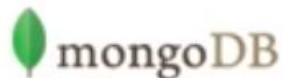
<https://www.docker.com/>

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.

Web Server



Database



Messaging



Orchestration



Libraries

Dependencies

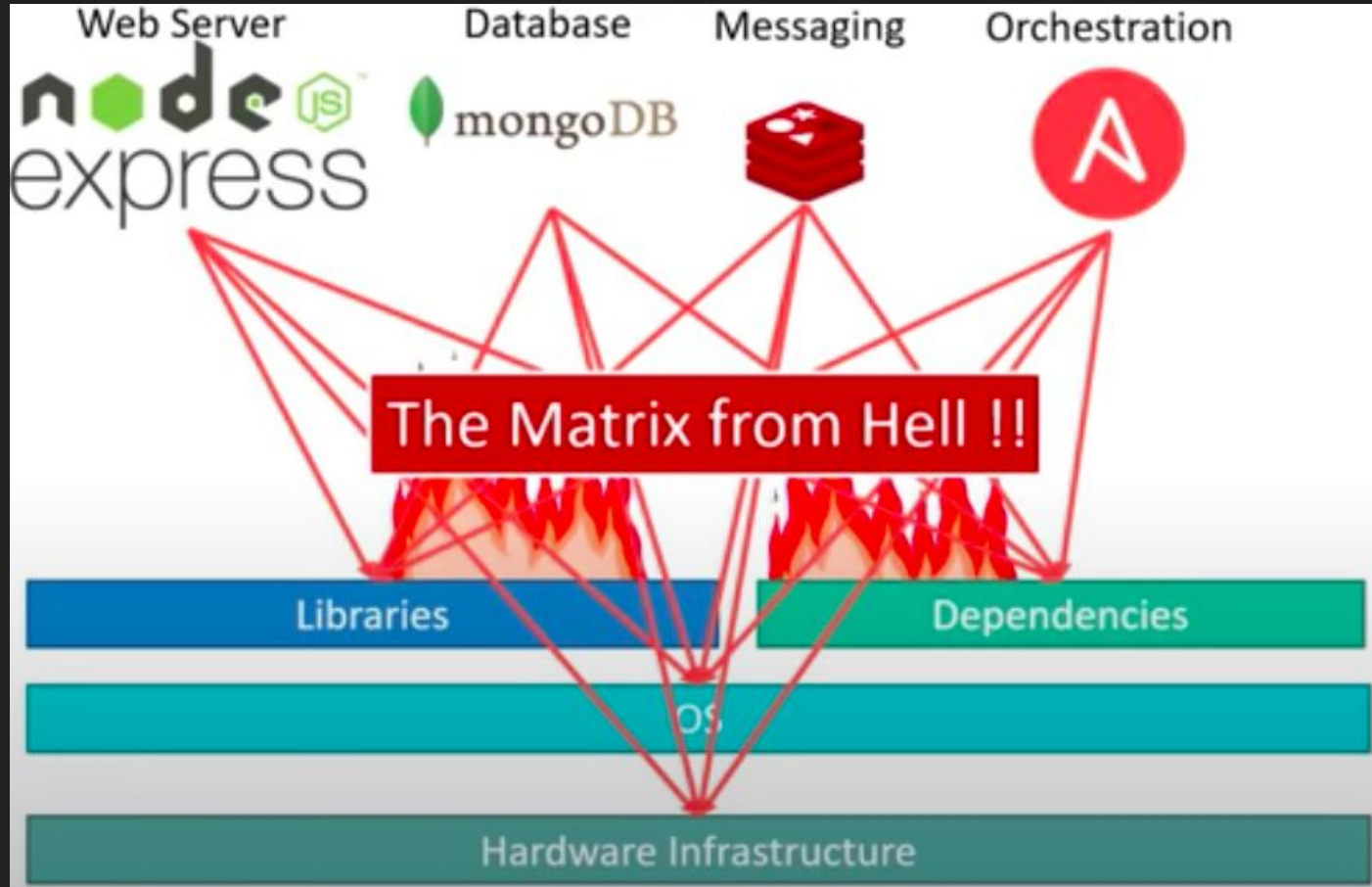
OS

Hardware Infrastructure

Compatibility/
Dependency

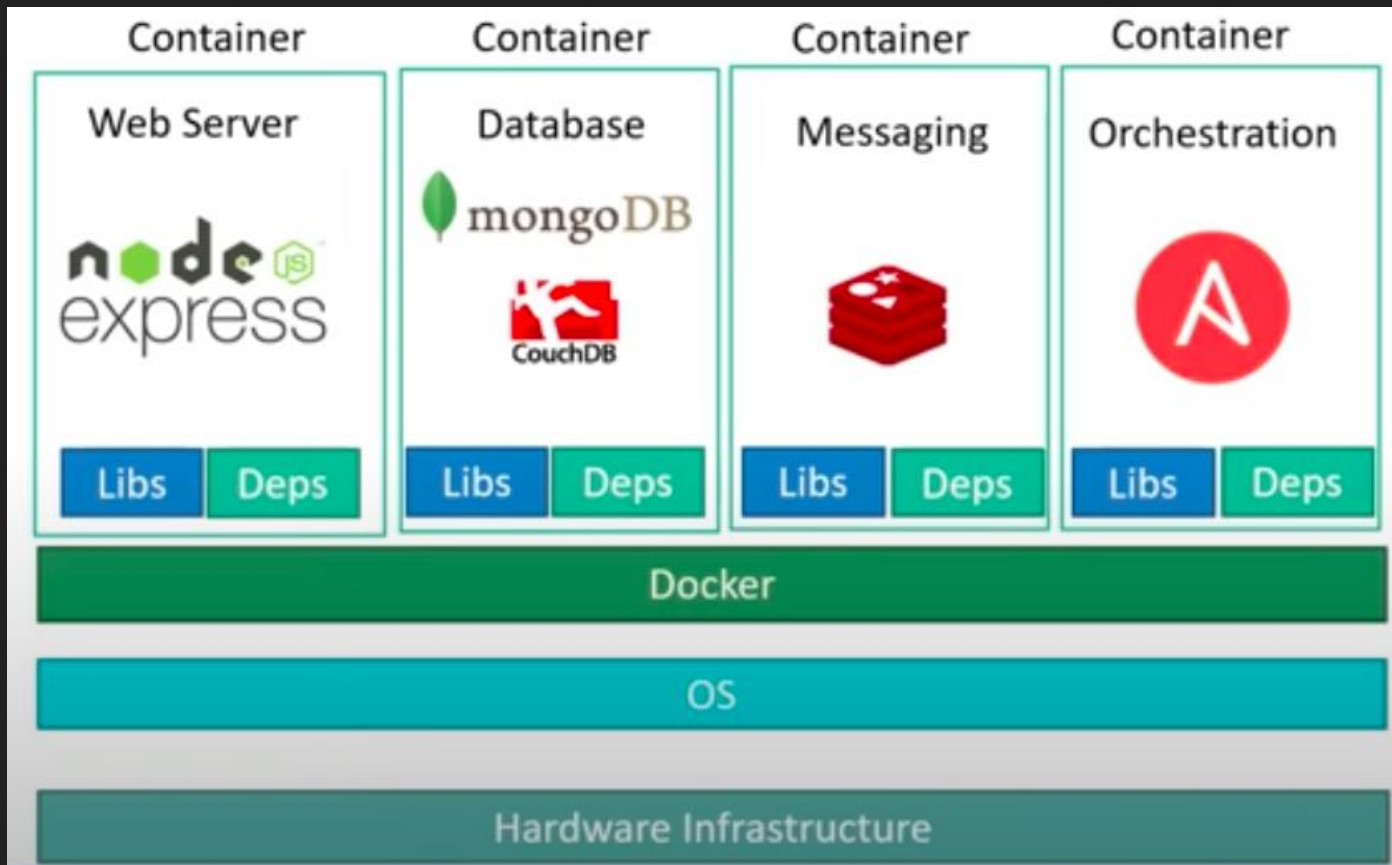
Long setup
time

Different
Dev/Test/Prod
environments

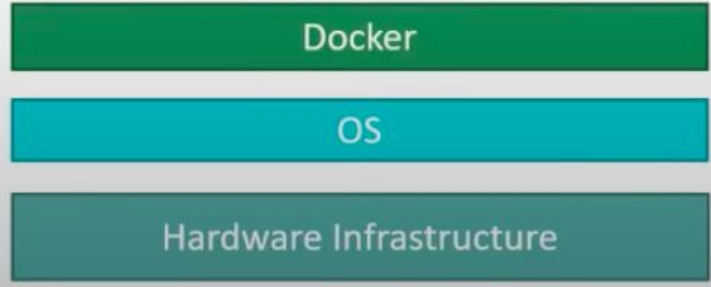
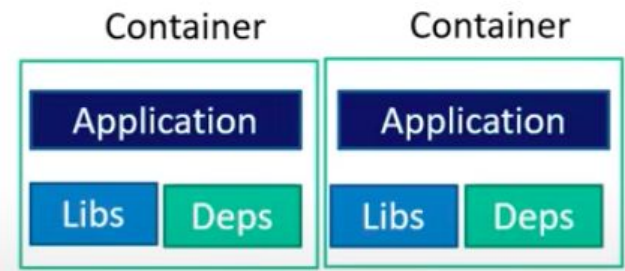
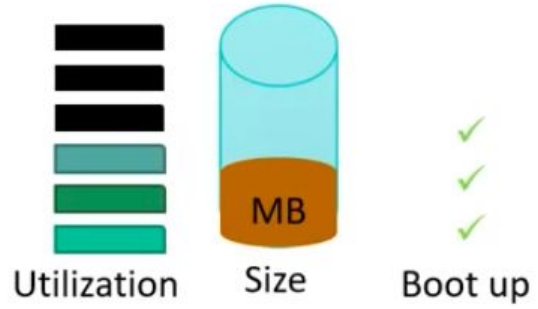
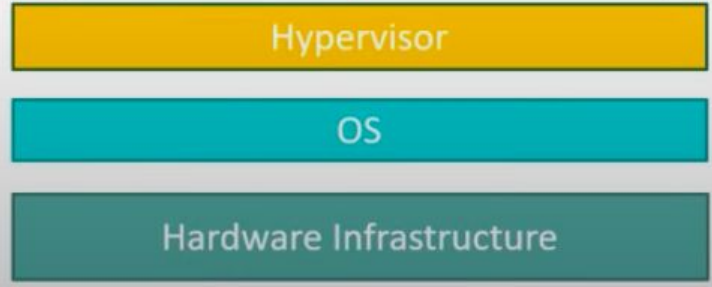
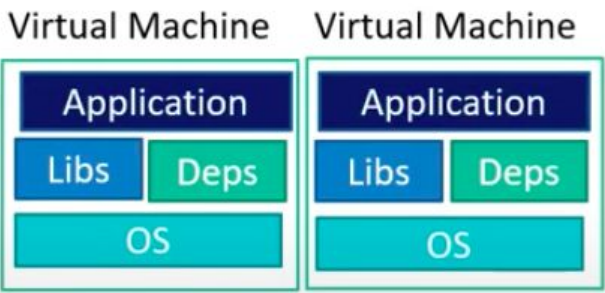
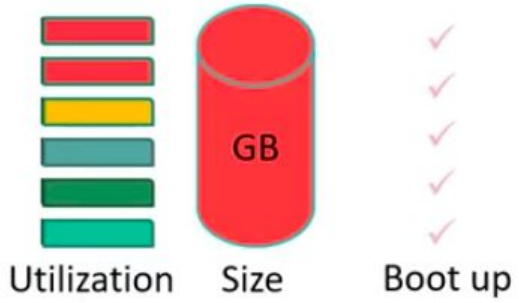


Containerized Application

Run each service with its own dependencies in separate containers



VMs vs Containers



From Application to Container



Developer



App.war



Guide



Operations



From Application to Container



Developer



App.war



DockerFile



Docker Image



Operations

It Fixes the traditional “but it works on my machine”

From Application to Container



Developer



Docker Image



Operations

It Fixes the traditional “but it works on my machine”

Docker

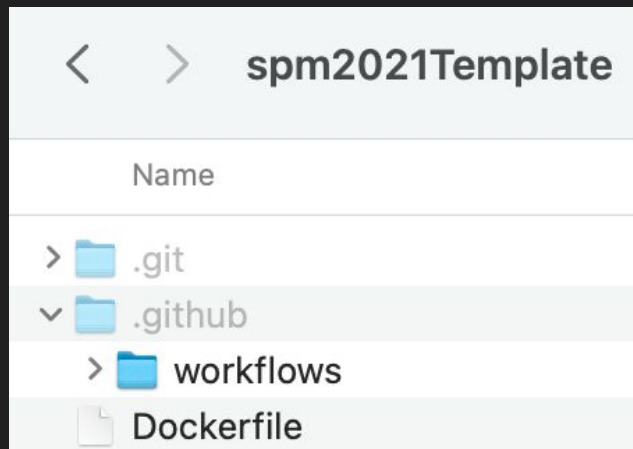
<https://docs.docker.com/get-docker/>

Dockerfile

```
FROM tomcat
```

```
COPY /target/spm2021.war /usr/local/tomcat/webapps/
```

```
CMD ["catalina.sh", "run"]
```

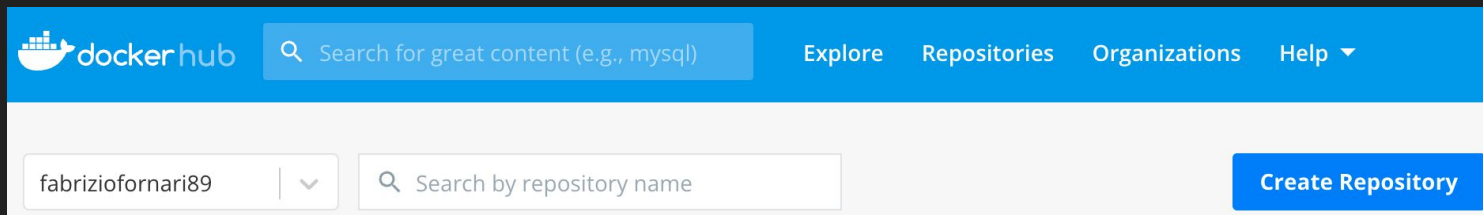


Public Docker Images Repository



<https://hub.docker.com/>

Create an account and a Private Repository



GitHub Actions

...

- name: Build with Maven
run: mvn -B package --file pom.xml test

- name: Build and Push Docker Image
uses: mr-smithers-excellent/docker-build-push@v5
with:
image: fabriziofornari89/spm2021template
registry: docker.io
username: \${{ secrets.DOCKER_USERNAME }}
password: \${{ secrets.DOCKER_PASSWORD }}

The screenshot shows the GitHub repository settings page for 'FabrizioFornari / spm2021template'. The 'Settings' tab is selected and circled in red. In the left-hand navigation menu, the 'Secrets' option is also circled in red. The main content area is titled 'Actions secrets' and contains the following information:

- Actions secrets** (with a 'New repository secret' button)
- Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions.
- Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#).
- Environment secrets** section: 'There are no secrets for this repository's environments.' Encrypted environment secrets allow you to store sensitive information, such as access tokens, in your repository environments. [Manage your environments and add environment secrets](#)
- Repository secrets** section: A table listing repository secrets.

Secret Name	Updated	Actions
DOCKER_PASSWORD	Updated 1 hour ago	Update Remove
DOCKER_USERNAME	Updated 1 hour ago	Update Remove

Docker Desktop

The screenshot shows the Docker Desktop interface. The top navigation bar is blue and contains the Docker logo, an 'Upgrade' button, and a user profile for 'fabriziofornari89'. The left sidebar has a menu with 'Containers / Apps', 'Images', 'Volumes', and 'Dev Environments' (with a 'PREVIEW' badge). The main content area is titled 'Images on disk' and shows '1 images' with a total size of '687.11 MB'. A progress bar indicates 'IN USE' and 'UNUSED' space. Below this, there are tabs for 'LOCAL' and 'REMOTE REPOSITORIES'. The 'REMOTE REPOSITORIES' tab is active, showing a dropdown menu for 'fabriziofornari89'. A table lists the images with columns for 'TAGS', 'OS', 'VULNERABILITIES', 'LAST PUSHED', and 'SIZE'. One image is listed: 'fabriziofornari89/spm2...' with tag 'master-38b44ff', OS 'linux', and a size of '369.49 MB'. The 'LAST PUSHED' column shows '30 minutes ago'. A 'Repositories per page' dropdown is set to '5'.

Containers / Apps

Images

Volumes

Dev Environments **PREVIEW**

Images on disk 1 images Total size: 687.11 MB **IN USE** UNUSED Clean up...

LOCAL REMOTE REPOSITORIES

fabriziofornari89 ▾

	TAGS	OS	VULNERABILITIES	LAST PUSHED	SIZE
🔒 fabriziofornari89/spm2...	master-38b44ff	linux	🛡️ Not available	30 minutes ago	369.49 MB

Repositories per page 5 ▾

Docker Desktop

Setup the Optional Settings so to specify the container name and the host port from which you will access the application

The screenshot shows the 'Optional Settings' dialog for a container named 'spm2021'. The container is based on the image 'my-spm2021-app:latest' and is currently in 'CREATION IN PROGRESS' with 'PORT: 8080'. The dialog is divided into three sections: 'Container Name', 'Ports', and 'Volumes'. In the 'Ports' section, the 'Local Host' field is set to '8080' and the 'Container Port' is '8080/tcp'. The 'Local Host' field is circled in red. At the bottom, there are 'Cancel' and 'Run' buttons.

spm2021 my-spm2021-app:latest
CREATION IN PROGRESS PORT: 8080

Optional Settings

Container Name
spm2021

Ports

Local Host	Container Port
8080	8080/tcp

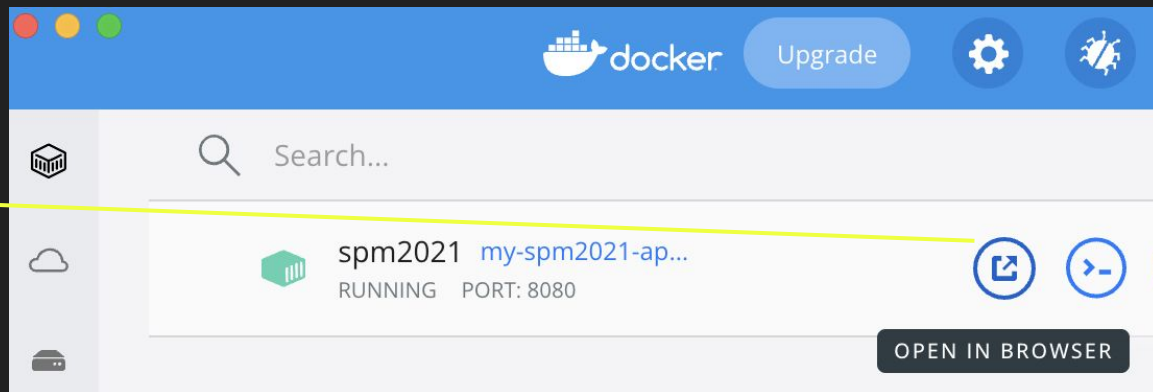
Volumes

Host Path	Container Path
...	

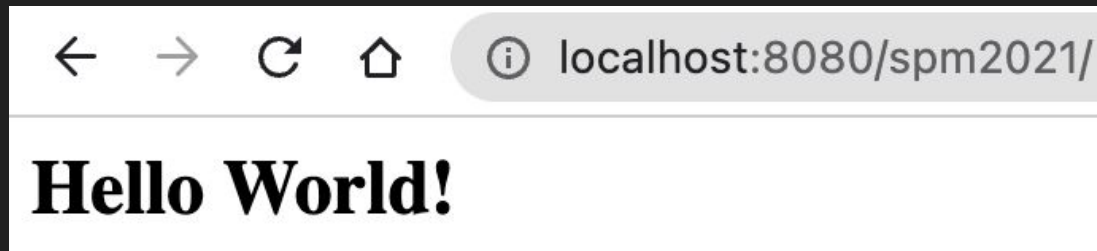
Cancel Run

Docker Desktop

Open the browser to the right address and port



`http://localhost:8080/spm2021` in my case



Jenkins + Docker



docker

jenkins

...so a Docker Host



What if a Docker Host fails?



Orchestrating Hosts



Orchestration technology focuses on clustering and managing containers and hosts.

Docker Swarm: Easy to setup but lacks autoscaling

Kubernetes: from Google, difficult to setup but supports many advanced features, all public cloud supports it

MESOS: from Apache, difficult to setup but supports many advanced features,

Kubernetes



A fundamental difference between Kubernetes and Docker is that Kubernetes is meant to run across a cluster while Docker runs on a single node. Kubernetes is more extensive than Docker Swarm and is meant to coordinate clusters of nodes at scale in production in an efficient manner.

What's next?

Date	Topic
10/12/2021	Sprint Review
16/12/2021	The role of Databases in CI/CD with a Special Guest
17/12/2021	Sprint Meeting/ Project Status Check
23/12/2021	Review of the Entire Course
January	Sprint Review