

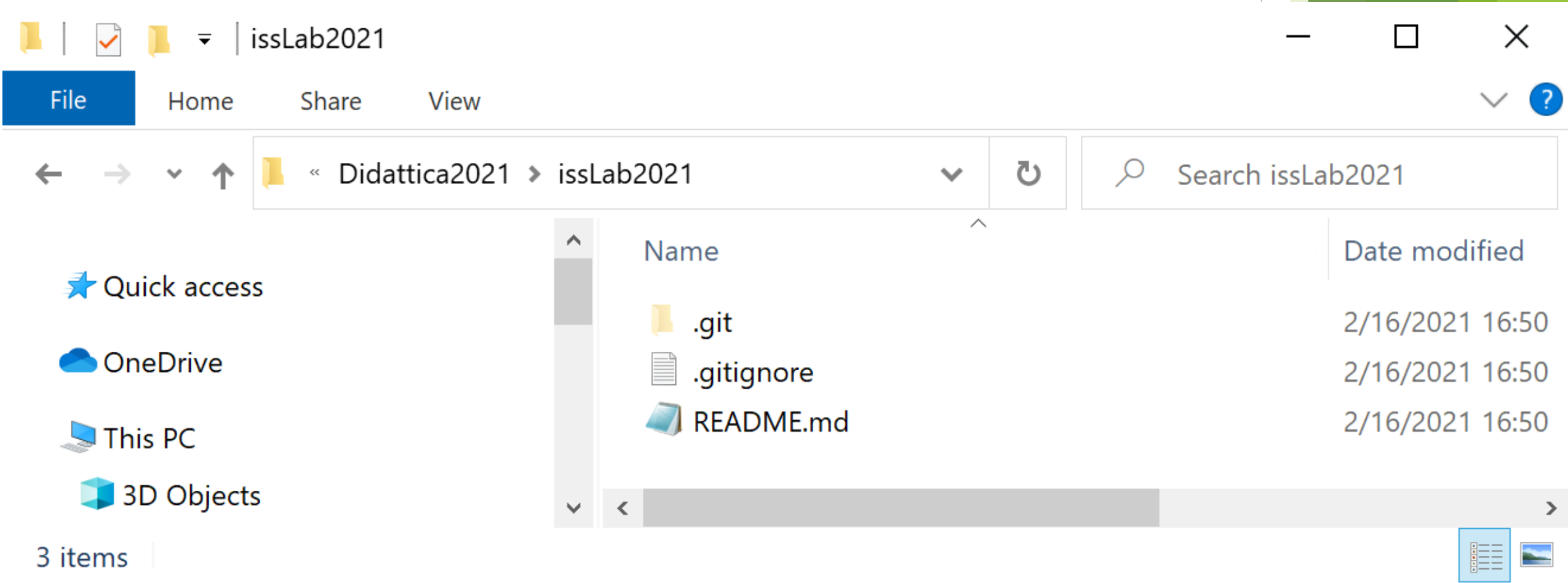
72939 - INGEGNERIA DEI SISTEMI SOFTWARE M

LABORATORIO

<https://www.unibo.it/it/didattica/insegnamenti/insegnamento/2020/385373>

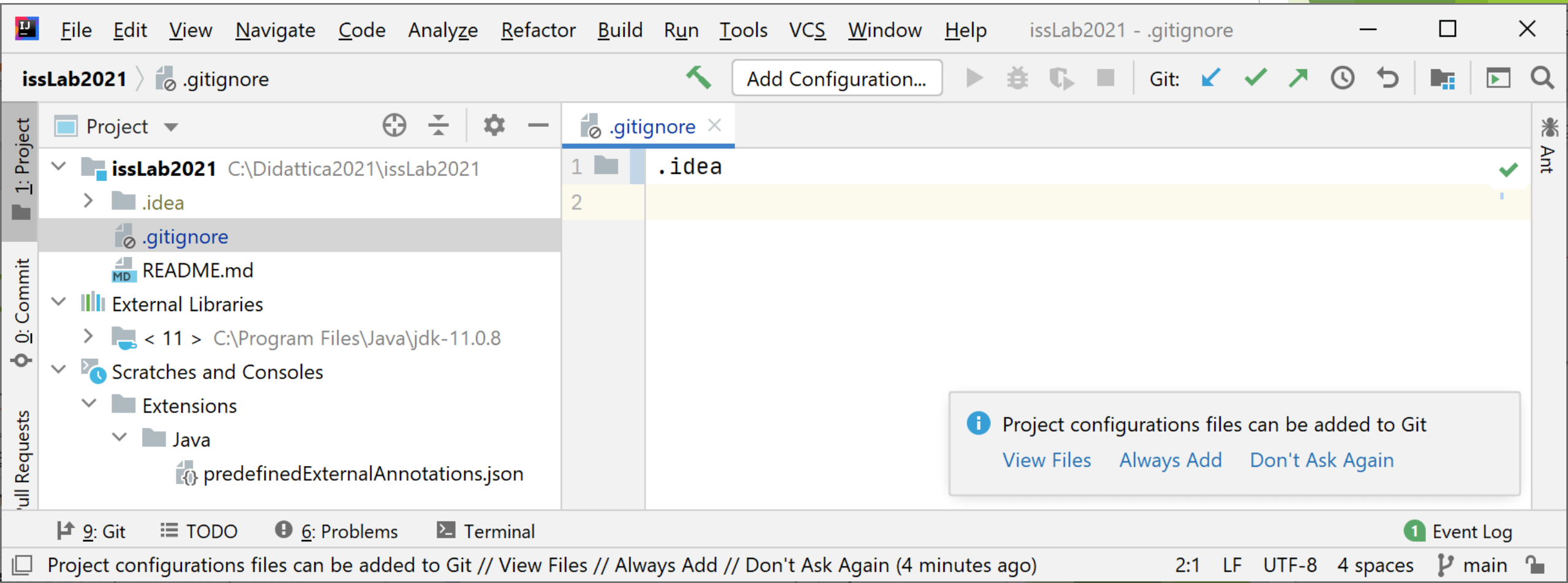
<https://github.com/anatali/issLab2021>

# Lab creation on git



# Open in IntelliJ

.gitignore created on github by selecting Java, then changed



# Copy a project (it.unibo.issLabStart)

The screenshot shows an IDE window titled "issLab2021 - .gitignore". The left sidebar displays the project structure for "issLab2021" located at "C:\Didattica2021\issLab2021". The project is expanded, showing the "it.unibo.issLabStart" sub-project. The "it.unibo.issLabStart" directory is highlighted, and its contents are listed: ".idea", ".settings", "Dockercompose", "virtualrobotandcontrol.yaml", "resources", "plugins" (containing "it.unibo.Qactork.ide\_1.2.4.jar", "it.unibo.Qactork.ui\_1.2.4.jar", and "it.unibo.Qactork\_1.2.4.jar"), "userDocs", ".classpath", ".gitignore", ".project", "it.unibo.issLabStart.iml", ".gitignore", and "README.md". The main editor area shows the ".idea" directory selected, with a list of files: ".idea" (line 1) and ".idea" (line 2). The right sidebar shows the "Ant" build system. The bottom status bar includes "Git", "TODO", "Problems", and "Terminal" tabs.



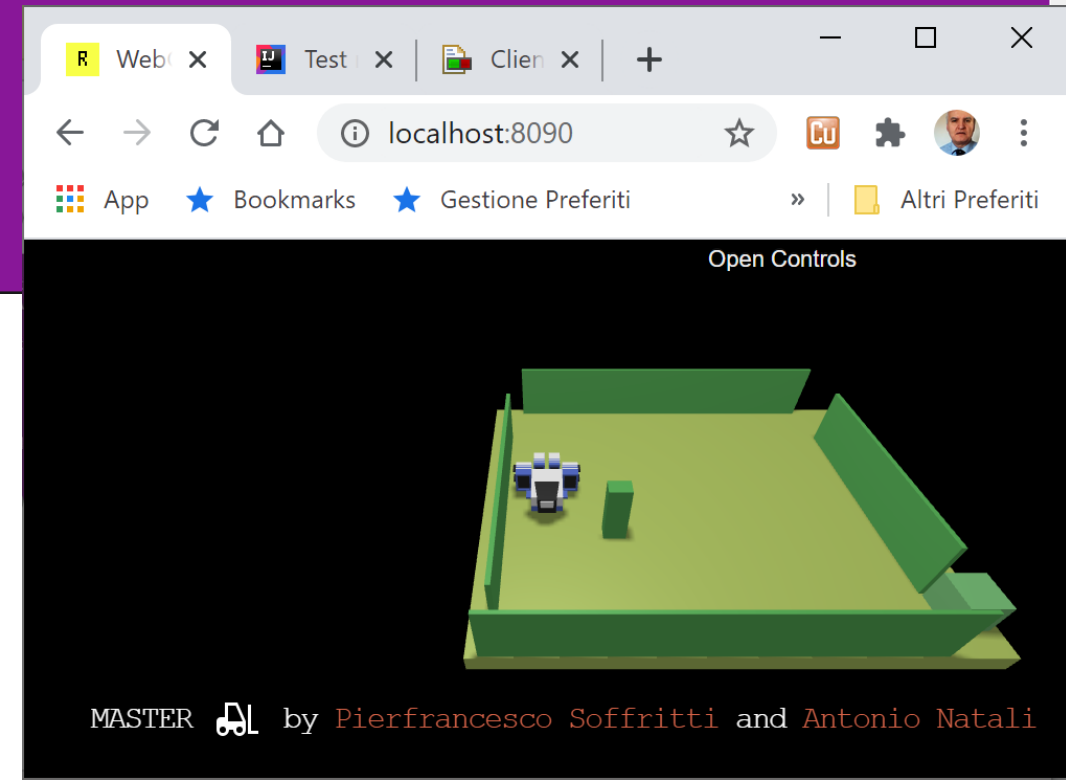
## Java Update Available

A new version of Java is ready to be installed!

# Run the WEnv

```
Viola - docker-compose -f virtualrobotandcontrol.yaml up wenv

C:\Didattica2021\issLab2021\it.unibo.issLabStart\Dockercompose>docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
C:\Didattica2021\issLab2021\it.unibo.issLabStart\Dockercompose>docker-compose -f virtualrobotandcontrol.yaml up wenv
Creating network "dockercompose_default" with the default driver
Creating dockercompose_wenv_1 ... done
Attaching to dockercompose_wenv_1
wenv_1              | WebpageServer WebGLScene | socketIndex=-1
wenv_1              | WebpageServer WebGLScene | connection socketIndex=0
wenv_1              | WebpageServer WebGLScene | MASTER-webpage ready
```



`docker-compose -f virtualrobotandcontrol.yaml up wenv`

localhost:8090

`docker-compose -f virtualrobotandcontrol.yaml down`

# Run the WEnv +

docker-compose -f virtualrobotandcontrol.yaml up  
Localhost:8090  
Localhost:3000

Viola - docker-compose -f virtualrobotandcontrol.yaml up

C:\Didattica2021\issLab2021\it.unibo.issLabStart\Dockecompose>docker-compose -f virtualrobotandcontrol.yaml up  
Creating network "dockercompose\_default" with the default driver  
Creating dockercompose\_wenv\_1 ... done  
Creating dockercompose\_vrobotcontrol\_1 ... done  
Attaching to dockercompose\_wenv\_1, dockercompose\_vrobotcontrol\_1  
wenv\_1 | WebpageServer WebGLScene | socketIndex=-1  
vrobotcontrol\_1 | server listening on port 3000 with \_\_dirname=/home/app  
wenv\_1 | \$\$\$ WebpageServer wssocket | client connected wssocketIndex=0  
vrobotcontrol\_1 | WebSocketClient | Connected  
wenv\_1 | WebpageServer WebGLScene | connection socketIndex=0  
wenv\_1 | WebpageServer WebGLScene | MASTER-webpage ready

Virtualrobot Controller

localhost:3000

App Bookmarks Gestione Preferiti Servizi IOT cultura Altri Preferiti

VirtualRobot-Console Selection

This *webguiServer* works on localhost:3000 and exploits *socket.io* to interact with the virtual environment server (*WEnv*) on port 8090.  
The server *WEnv* sends over *socket.io* information about the current state of the virtual world.

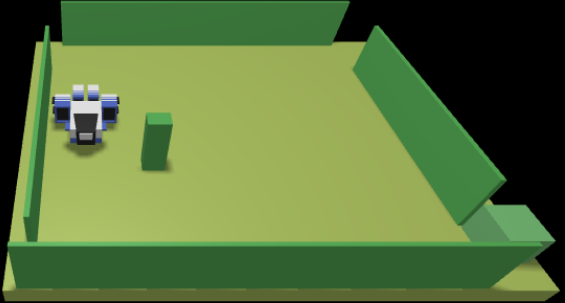
Use the simple, conventional gui Use the JQuery gui Use the websocket gui


Show Connections ClearHistory

Move time: (double-click to send)

DISPLAY AREA

Open Controls



MASTER  by Pierfrancesco Soffritti and Antonio Natali

## Requirement

Build an application that moves the virtual robot

## TestPlan

Check that the requested move has been completed with success or that it is failed

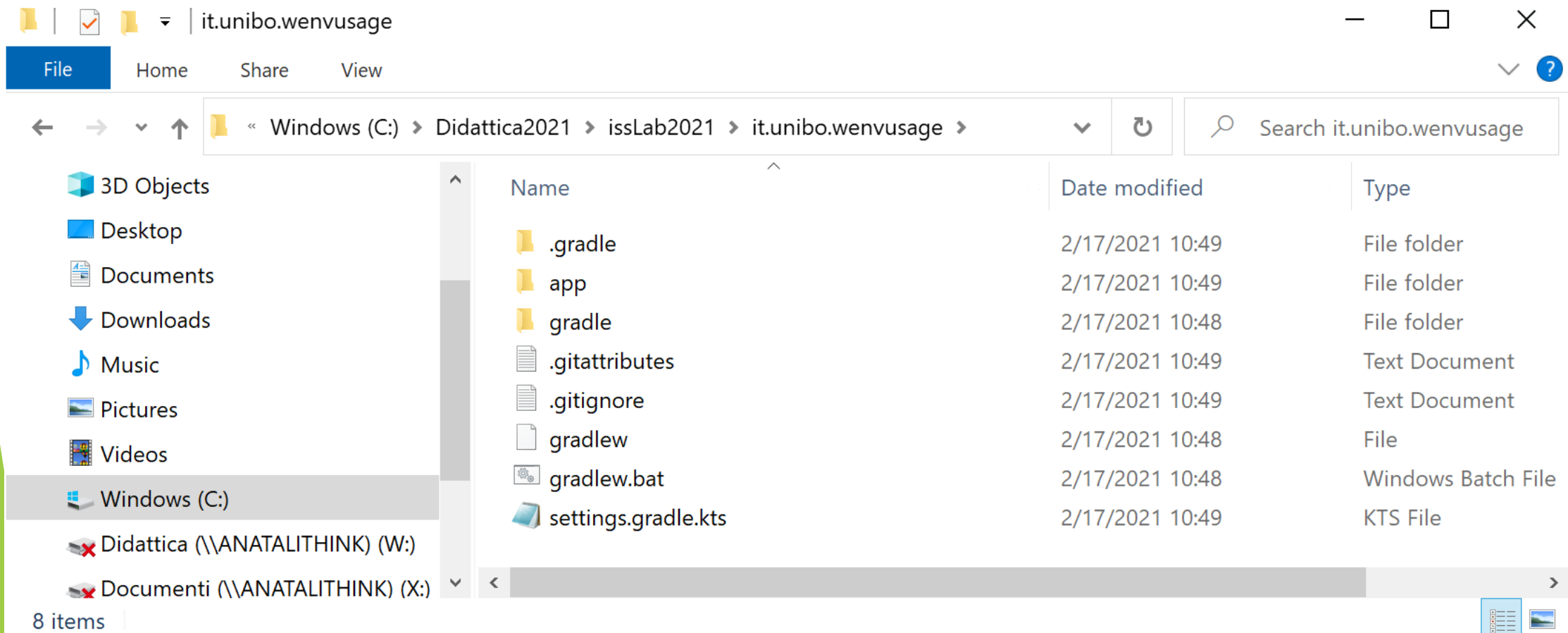
## Analysis

**TODO:** (*HINT: focalize on the relevant aspects*)

# Moving the virtual robot

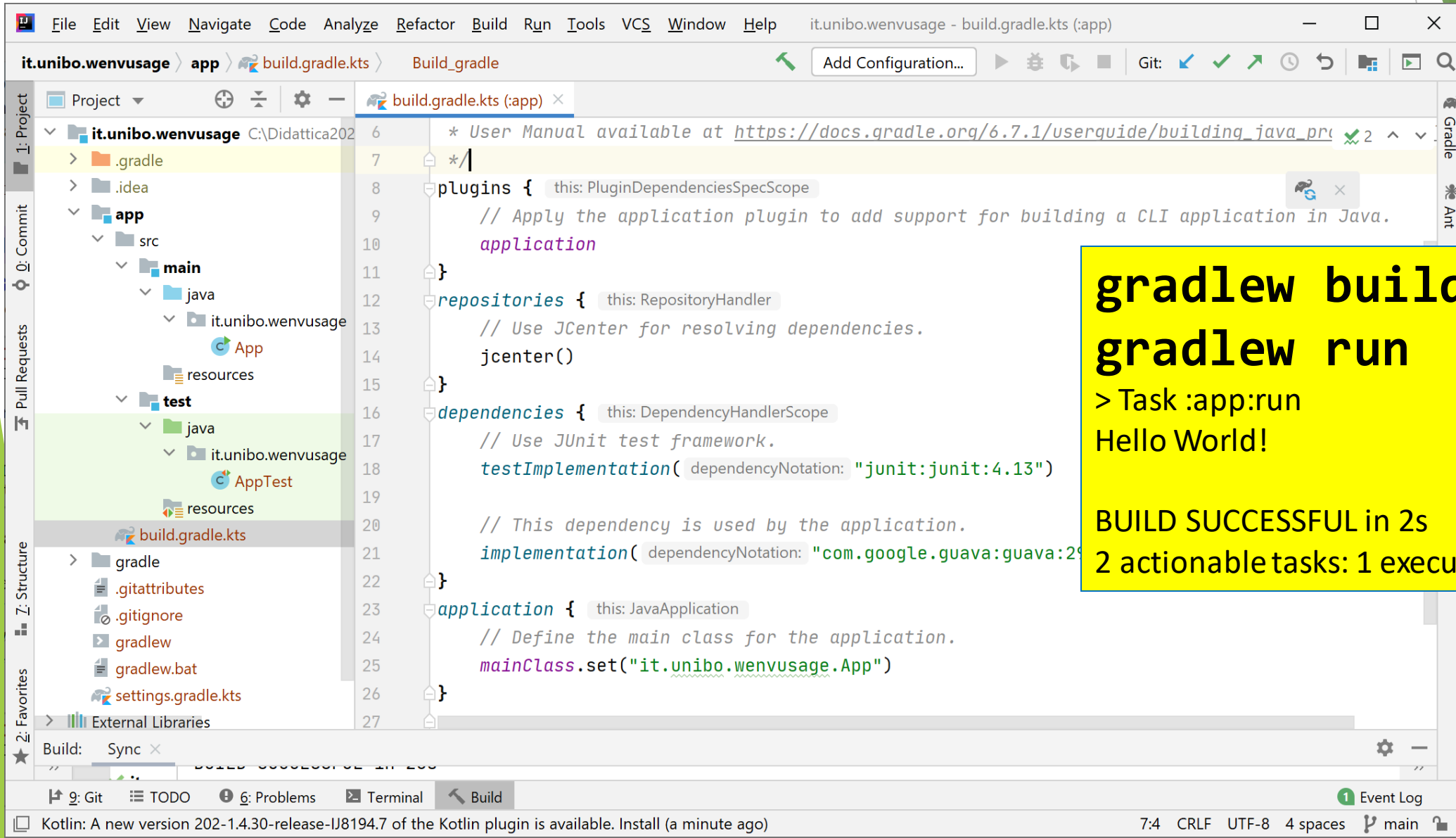
```
C:\...\issLab2021>mkdir it.unibo.wenvusage
```

```
C:\...\issLab2021>gradle init (select 2, 3, 1, 2, 1, demo, demo)
```





# The project it.unibo.wenvusage



The screenshot shows the IDE interface with the `build.gradle.kts` file open. The left sidebar displays the project structure, including the `src/main/java` directory containing the `it.unibo.wenvusage` package and the `App` class. The main editor area shows the following code:

```
6  * User Manual available at https://docs.gradle.org/6.7.1/userguide/building\_java\_projects.html ✓ 2 ^ v
7  */
8  plugins { this: PluginDependenciesSpecScope
9      // Apply the application plugin to add support for building a CLI application in Java.
10     application
11 }
12 repositories { this: RepositoryHandler
13     // Use JCenter for resolving dependencies.
14     jcenter()
15 }
16 dependencies { this: DependencyHandlerScope
17     // Use JUnit test framework.
18     testImplementation( dependencyNotation: "junit:junit:4.13")
19
20     // This dependency is used by the application.
21     implementation( dependencyNotation: "com.google.guava:guava:29.0-jre")
22 }
23 application { this: JavaApplication
24     // Define the main class for the application.
25     mainClass.set("it.unibo.wenvusage.App")
26 }
27
```

The bottom status bar indicates the build was successful in 2s, with 2 actionable tasks: 1 executed, 1 up-to-date.

**gradlew build**  
**gradlew run**

> Task :app:run  
Hello World!

**BUILD SUCCESSFUL** in 2s  
2 actionable tasks: 1 executed, 1 up-to-date

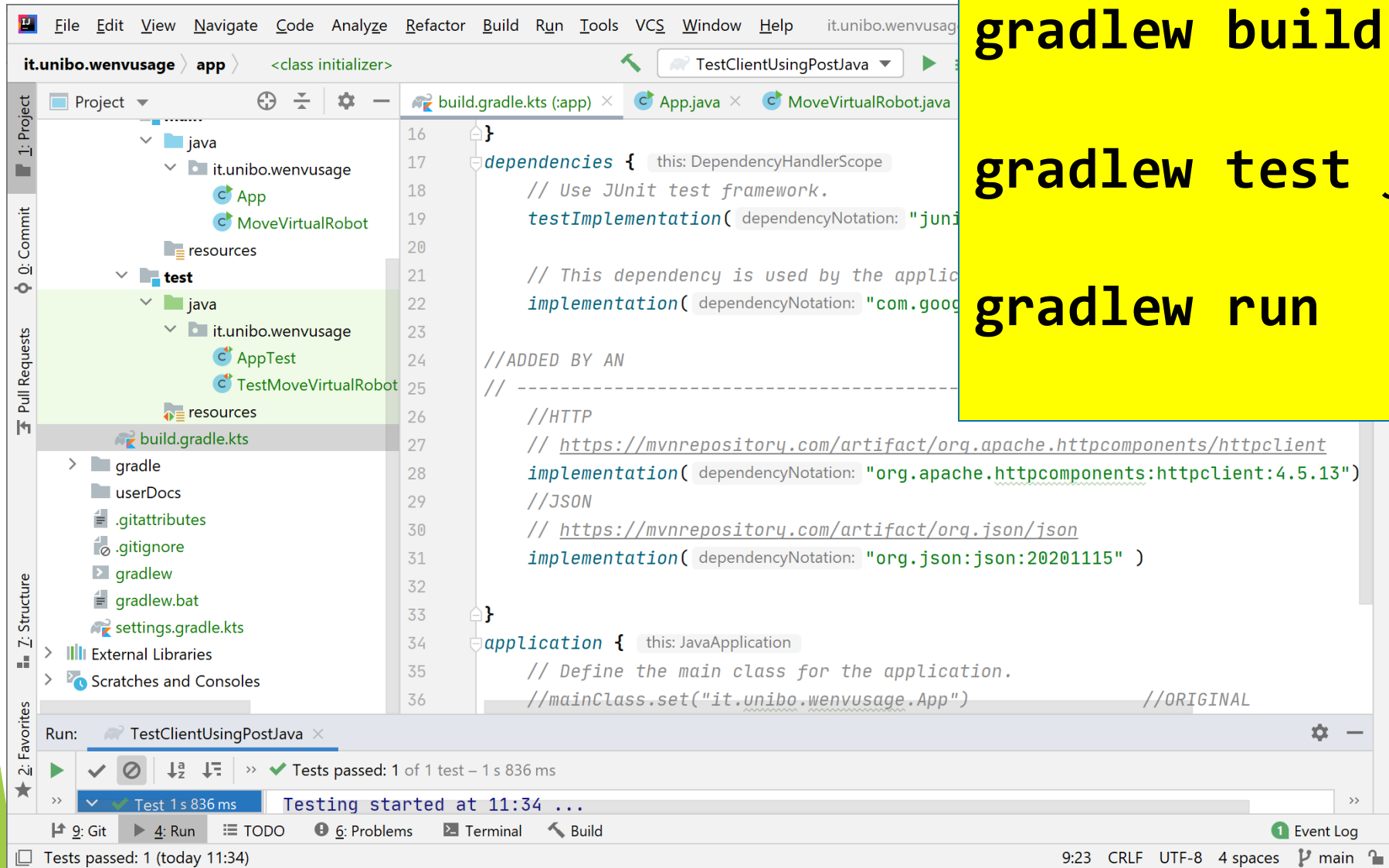
## Define our app and modify build

```
application {  
    // Define the main class for the application.  
    //mainClass.set("it.unibo.wenvusage.App")    //ORIGINAL  
    mainClass.set("it.unibo.wenvusage.MoveVirtualRobot")  
}
```

Create the **userDocs** folder

Formally define the Test (in JUnit)

# The project it.unibo.wenvusage



**gradlew build -x test**

**gradlew test jacocoTestReport**

**gradlew run**

