

## Test management tools Jenkins Tutorial

### Table of Contents

1. Access Jenkins .....	2
2. Connect to Jenkins .....	2
3. Create a new Job .....	2
4. Configure your Job .....	3
5. Build .....	<b>Error! Bookmark not defined.</b>
6. Sending results from Jenkins to Testlink .....	<b>Error! Bookmark not defined.</b>
7. View the state of a Test case in Testlink .....	<b>Error! Bookmark not defined.</b>

## 1. Access Jenkins

<https://www.scs.ubbcluj.ro:9090>

## 2. Connect to Jenkins

### Log in - user your scs\_id\_user

user: for example, for student Savu Victor Gabriel (**svie1164@scs.ubbcluj.ro**) the user is **svie1164**

password: the password of user **svie1164**

## 3. Create a new Job

Each student will create his/her own **Job (one for each TestPlan created in Testlink)**, the name of the job will be the scs id user concatenated with the word Job

- for example, for student Savu Victor Gabriel (**svie1164@scs.ubbcluj.ro**) the name of the Test Plans will be: **svie1164Job\_BBT**, **svie1164Job\_WBT**, **svie1164Job\_IntegrationT**

### Create Job - STEPS

- Jenkins Menu
- Select **New Job**
  - Name:
    - scs\_user\_idJob\_BBT
    - scs\_user\_idJob\_WBT
    - scs\_user\_idJob\_IntegrationT
  - Select *Build a free -style software project*
  - click button "ok"

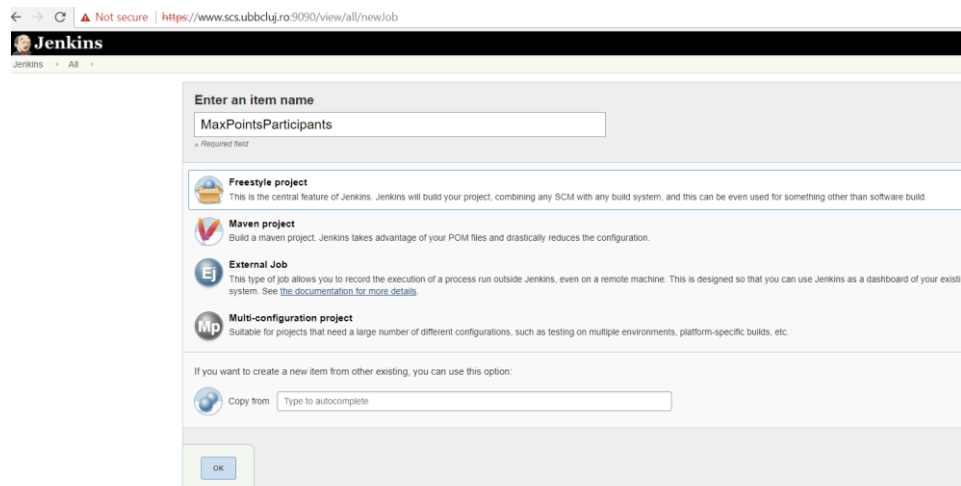


Figure 1. Jenkins – Creating new job

## 4. Configure your Job

- Jenkins Menu
- Select from the list of Jobs your previously created job
- Select from left menu the "Configure" section

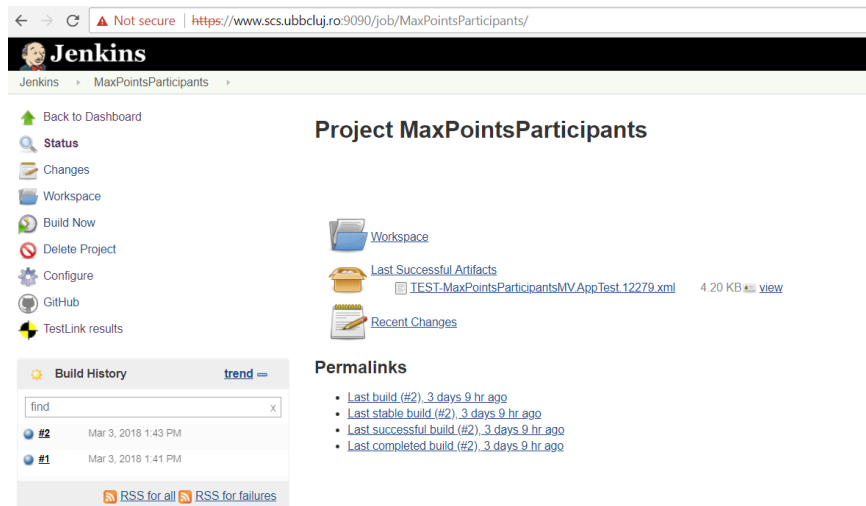
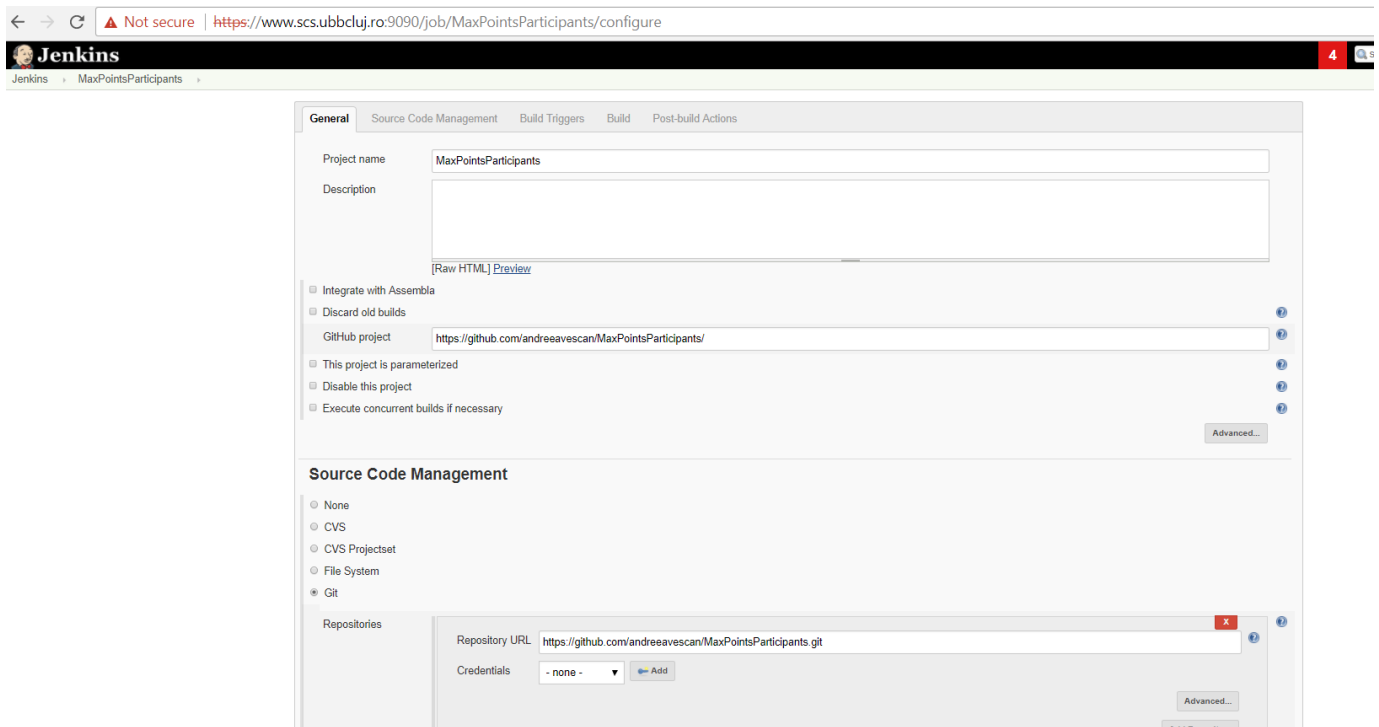
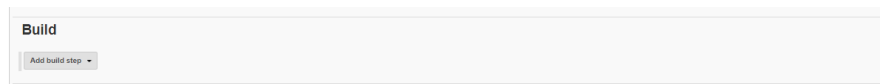


Figure 2. Jenkins – Configure job

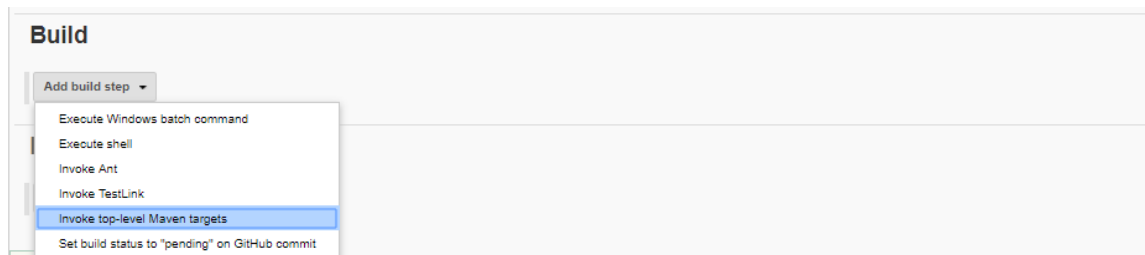
- GitHub project and Repository URL



- From Build section



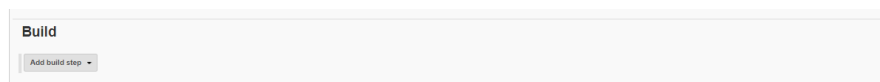
- Select Invoke top-level Maven targets



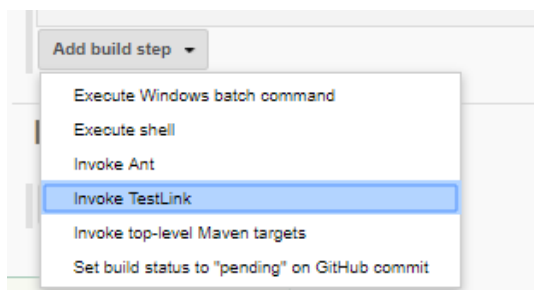
- Add info: Maven version: mvn and Goals: compile



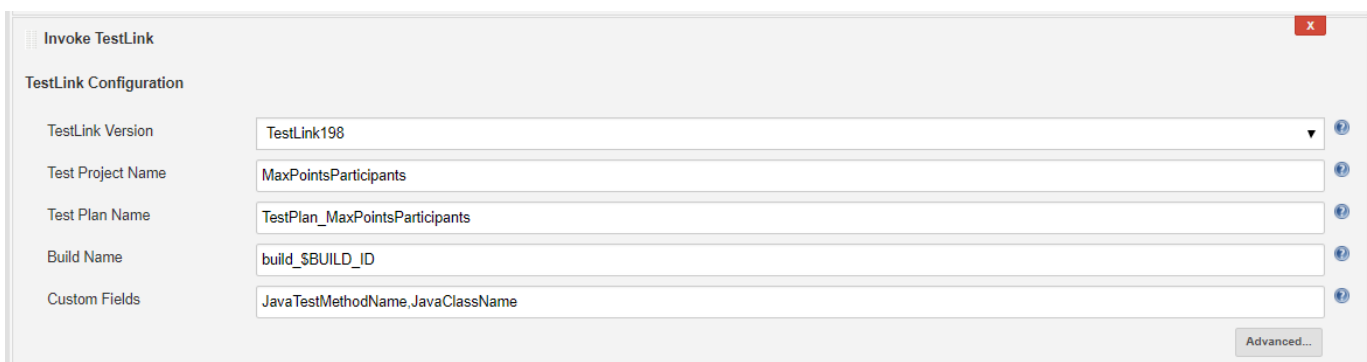
- From Build section



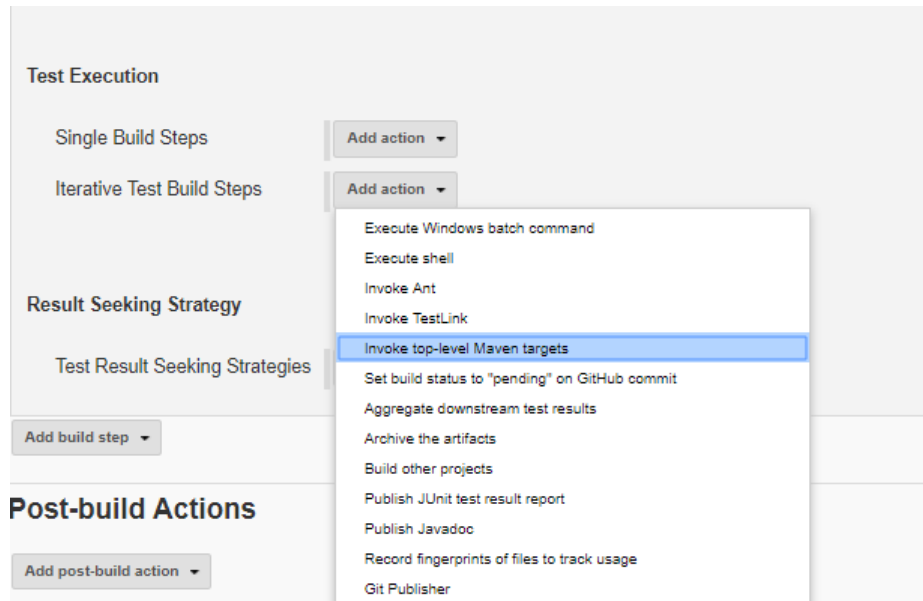
- Select Invoke TestLink



- Add info: Your TestProjectName, your Test Plan Name



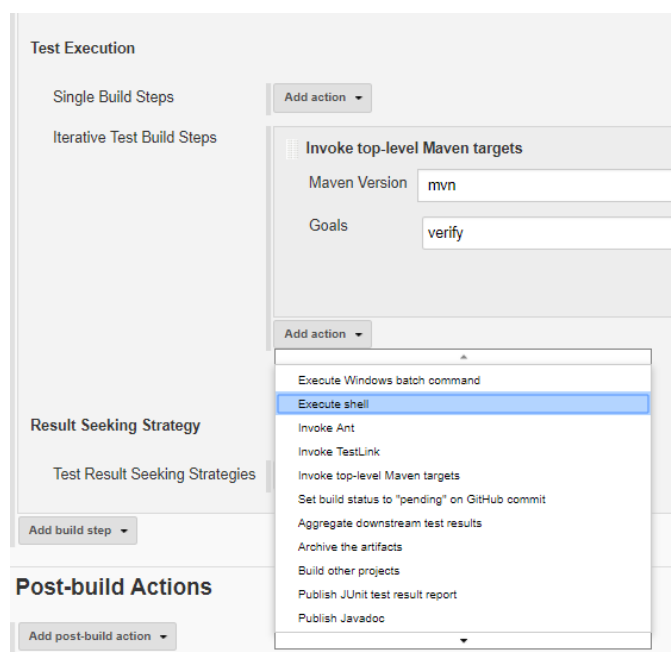
- On Invoke TestLink, at Test execution, on Iterative Test Build Steps, select an action



- Select Invoke top-level Maven targets and add the information: Maven: mvn and Goals: verify

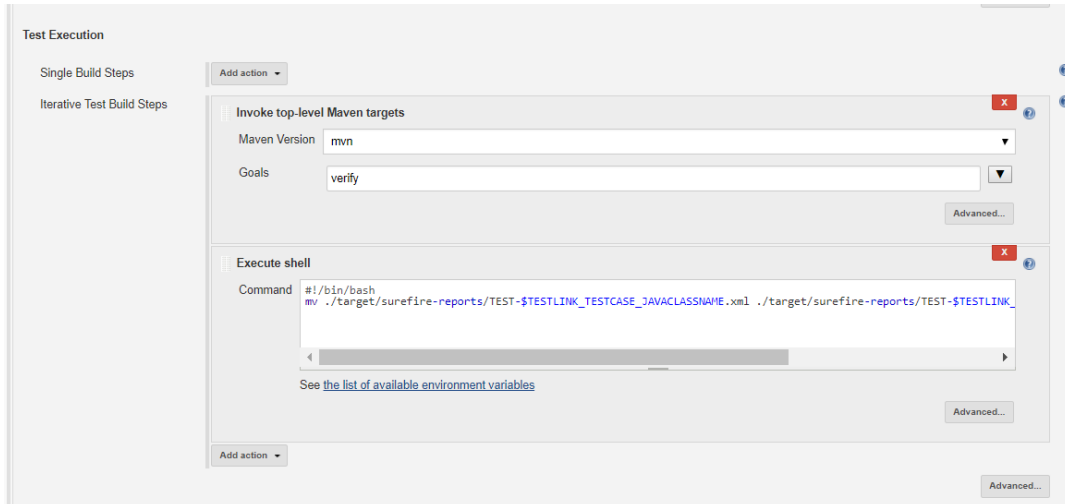


- Add another action: Execute Shell



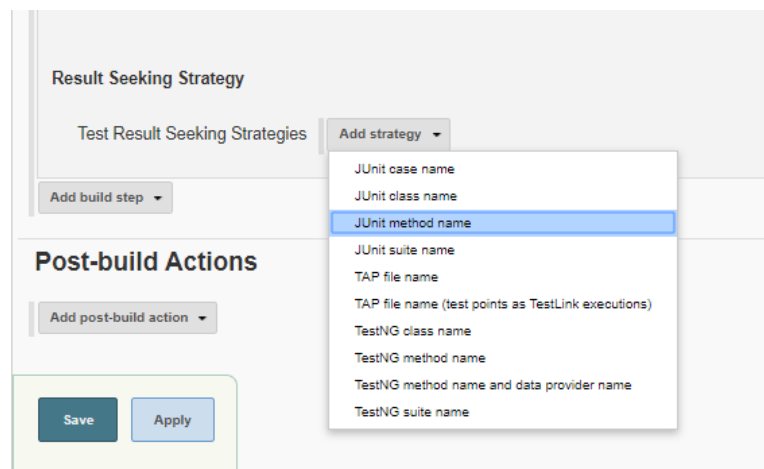
- Add the following command

```
#!/bin/bash
mv ./target/surefire-reports/TEST-$TESTLINK_TESTCASE_JAVACLASSNAME.xml ./target/surefire-reports/TEST-$TESTLINK_TESTCASE_JAVACLASSNAME.$TESTLINK_TESTCASE_ID.xml
```



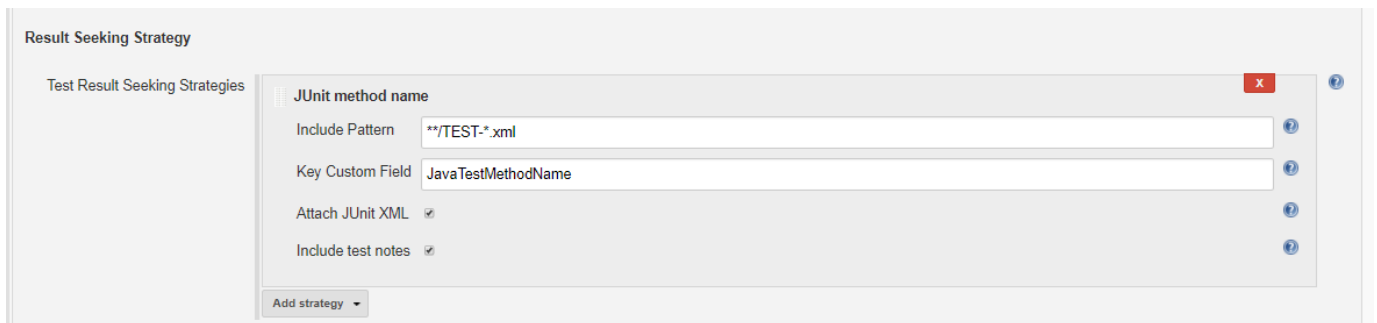
The screenshot shows the 'Test Execution' configuration page in Jenkins. On the left, there are tabs for 'Single Build Steps' and 'Iterative Test Build Steps'. The 'Iterative Test Build Steps' tab is active, showing two build steps: 'Invoke top-level Maven targets' and 'Execute shell'. The 'Invoke top-level Maven targets' step has 'Maven Version' set to 'mvn' and 'Goals' set to 'verify'. The 'Execute shell' step has a command field containing the shell script from the previous block. Both steps have an 'Advanced...' button to the right. At the bottom, there is an 'Add action' button and another 'Advanced...' button.

- On Invoke TestLink, at Result Seeking Strategy, add the strategy “JUnit method name”



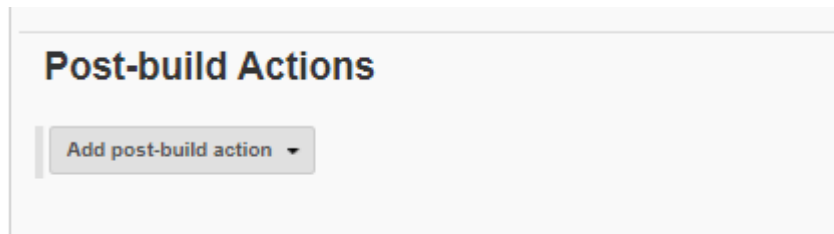
The screenshot shows the 'Result Seeking Strategy' configuration page. It has a section for 'Test Result Seeking Strategies' with an 'Add strategy' button. Below this is a section for 'Post-build Actions' with an 'Add post-build action' button. At the bottom are 'Save' and 'Apply' buttons. A dropdown menu is open, showing a list of strategies: 'JUnit case name', 'JUnit class name', 'JUnit method name' (which is highlighted), 'JUnit suite name', 'TAP file name', 'TAP file name (test points as TestLink executions)', 'TestNG class name', 'TestNG method name', 'TestNG method name and data provider name', and 'TestNG suite name'.

- Add the following information

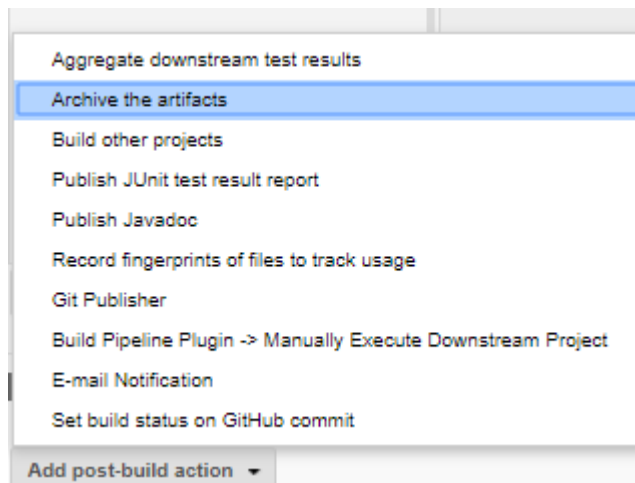


The screenshot shows the 'Result Seeking Strategy' configuration page with the 'JUnit method name' strategy selected. The configuration fields are: 'Include Pattern' set to '\*\*/TEST-\*.xml', 'Key Custom Field' set to 'JavaTestMethodName', 'Attach JUnit XML' checked, and 'Include test notes' checked. There is an 'Add strategy' button at the bottom left.

- At section “Post-build Actions”



- At section “Post-build Actions”, add the post-build action “Archive the artifacts”



- Add the information



- And Save