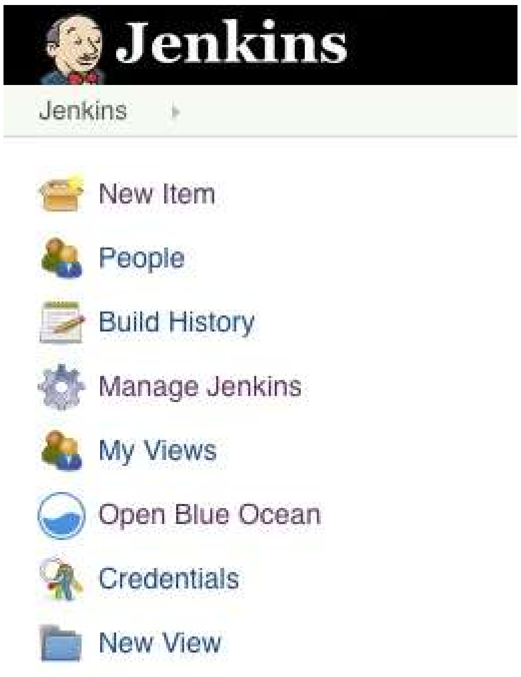
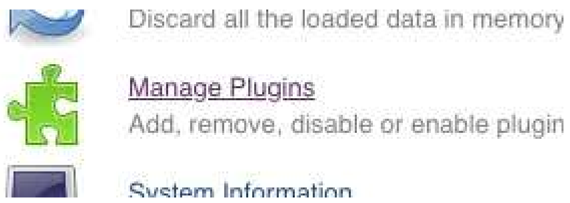


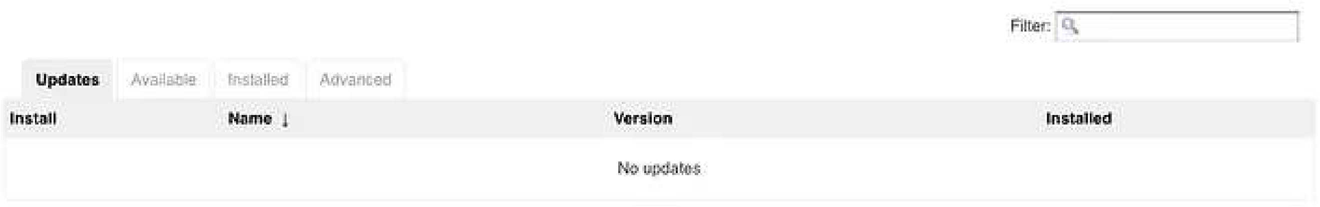
Jenkins menu

Click in **Manage Plugins:**


Discard all the loaded data in memory
Manage Plugins
Add, remove, disable or enable plugin
System Information

Manage Plugins

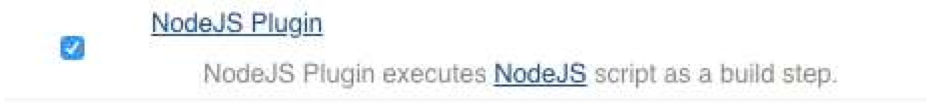
In Plugin manager, click on **Available** tab and after load the tab, search the plugins.



Install	Name ↓	Version	Installed
		No updates	

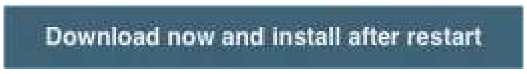
Installation tabs

Search for **nodejs** and mark the checkbox.



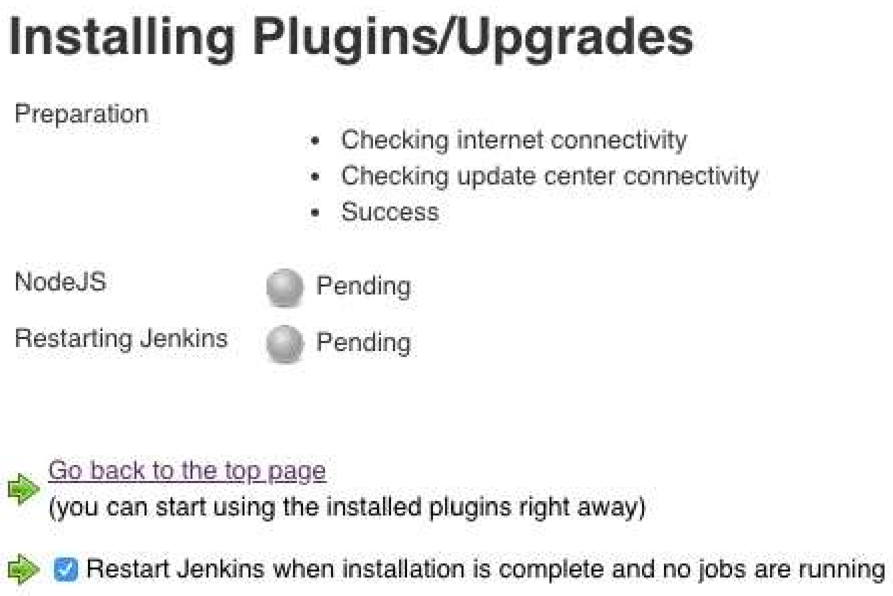
NodeJS plugin checked

Click on button "Download now and install after restart".



Download now and install after restart button

On next screen, mark the checkbox “Restart Jenkins when installation is complete and no jobs are running”.



Installing Plugins/Upgrades

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

NodeJS ● Pending

Restarting Jenkins ● Pending

Go back to the top page
(you can start using the installed plugins right away)

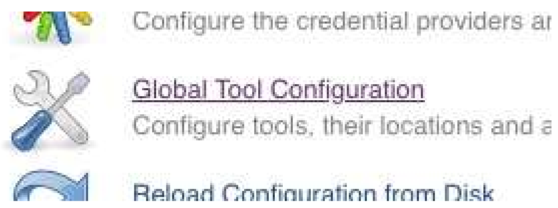
☑ Restart Jenkins when installation is complete and no jobs are running

Installating Plugins with Restart checked

Wait a moment, if the screen not change, go to jenkins home clicking in jenkins’s logo on top, in the left side of the site.

Configuring Node.js tool

After installed and restarted, go to jenkins’s home > Manage Jenkins > Global Tool Configuration.



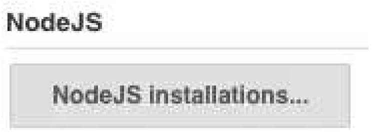
Configure the credential providers ar

Global Tool Configuration
Configure tools, their locations and a

Reload Configuration from Disk

Global Tool Configuration

Search for NodeJS and click on NodeJS instalation button.

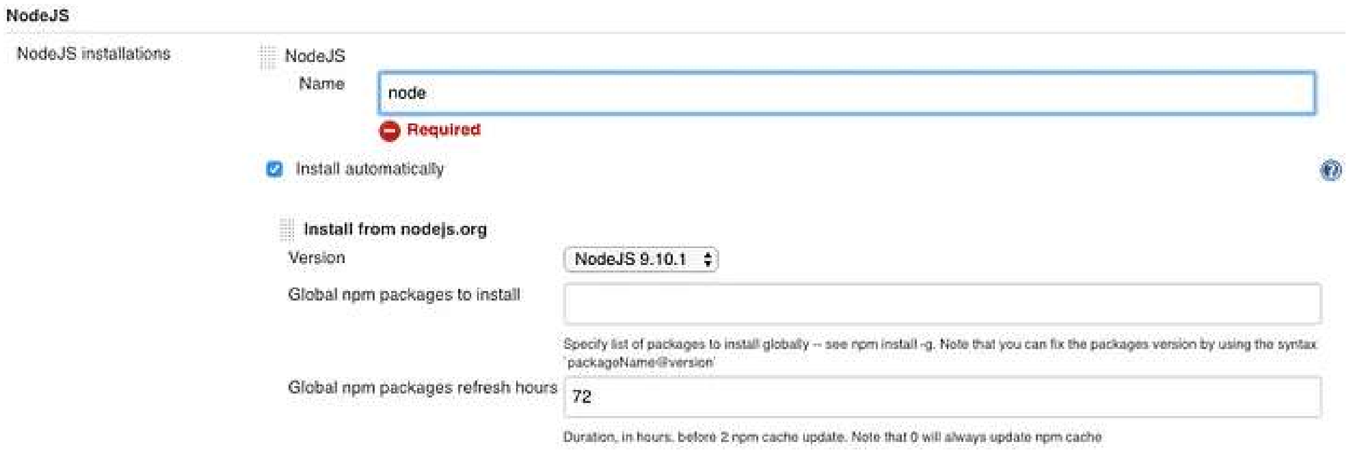


NodeJS

NodeJS installations...

NodeJS Installations button

Put a name for node configuration, ex: **node**. Select the version of node that you need.


NodeJS

NodeJS installations

NodeJS

Name: node

Required

[✓] Install automatically

Install from nodejs.org

Version: NodeJS 9.10.1

Global npm packages to install:

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

Global npm packages refresh hours: 72

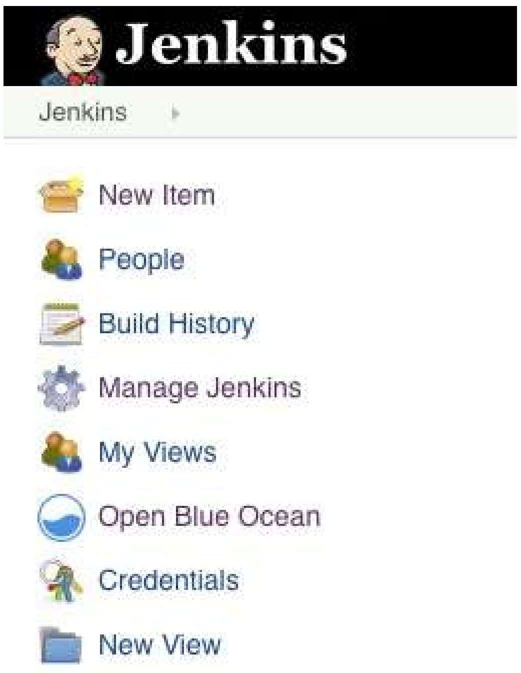
Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

NodeJS config

Test the nodejs plugin

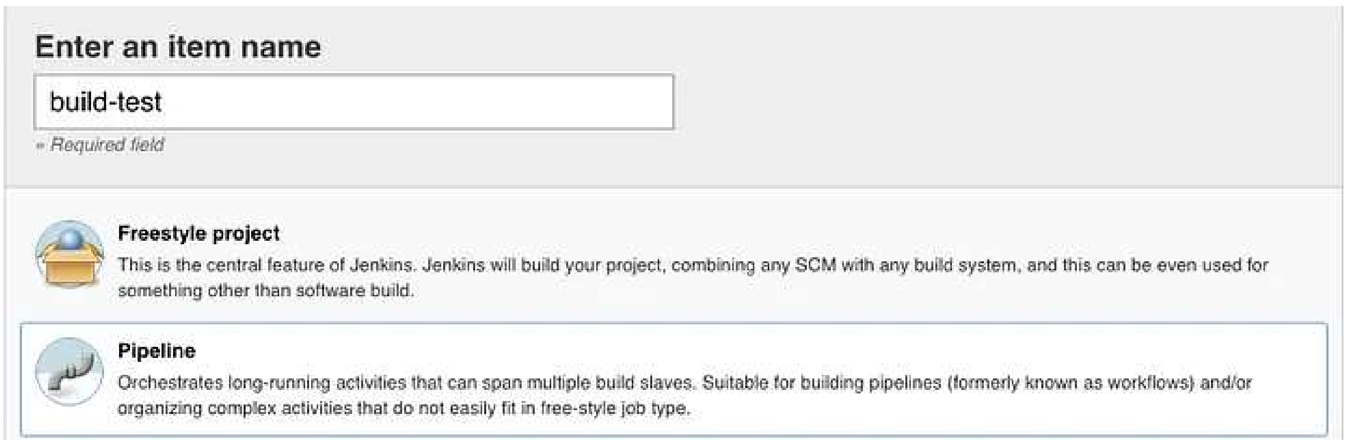
OK. NodeJS is installed with sucess. Let’s try it.

Go to jenkins’s homne, and click on **New Item** on jenkins’s menu.



Jenkins menu

Let’s create a job. Name it with **build-test** and select Pipeline. Click on “OK”.


Enter an item name

build-test

» Required field

Freestyle project

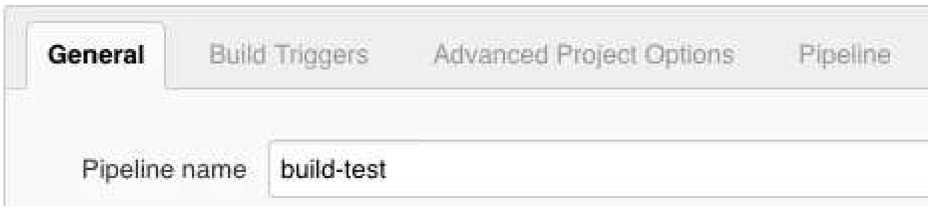
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline

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

creating job

You will see this:



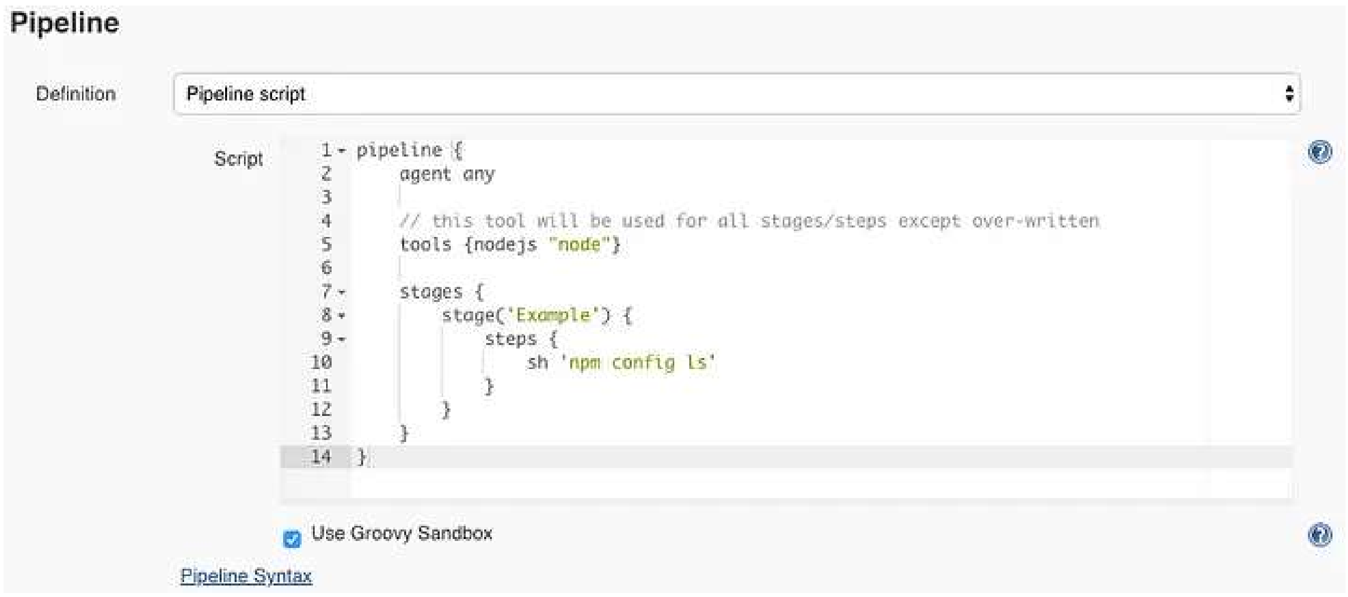
Pipeline name: build-test

job tabs

In pipeline, past this code:

```
pipeline {
  agent any

  tools {nodejs "node"}

  stages {
    stage('Example') {
      steps {
        sh 'npm config ls'
      }
    }
  }
}
```

The job will be like this:



Definition: Pipeline script

Script:

```
pipeline {
    agent any

    // this tool will be used for all stages/steps except over-written
    tools {nodejs "node"}

    stages {
        stage('Example') {
            steps {
                sh 'npm config ls'
            }
        }
    }
}
```

Pipeline Script

Pipeline explanation

Before continue, let's understand the pipeline job:

```
agent any
```

The job will run in any jenkins agent.

- You can have a lot of jenkins nodes, one master and some slaves, and the job will run in any node.
- You can run the job on specific jenkins, ex: You have 3 jenkins with Linux and one with Windows. You can run this job only in jenkins slave with Windows.

```
tools {nodejs "node"}
```

This line search for nodejs tool named “node” and use it in pipeline. It’s necessary for next scripts find the commands.

```
stages {
```

Stages is the inicialization from pipeline steps.

```
stage('Example') {
  steps {
    ...
  }
}
```

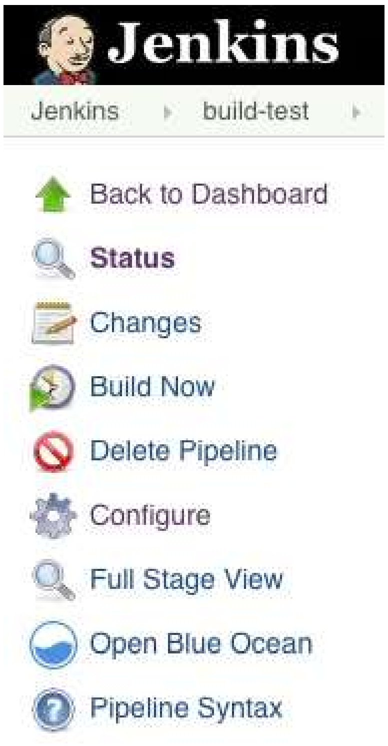
This commands start a new stage named “Example” and run the steps inside this stage.

```
sh 'npm config ls'
```

It execute a sh script in machine. With this, we can test the npm running correctly.

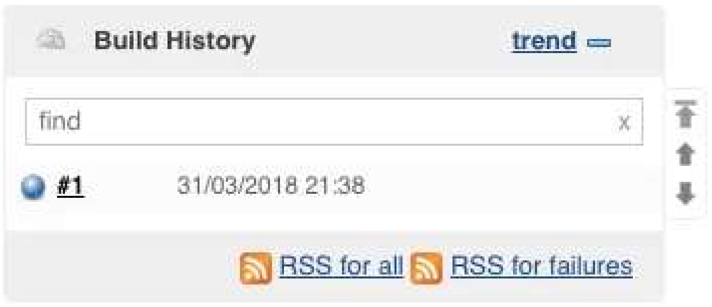
Running the test

Now, let’s run the script. Save the job. Inside the job, in left menu, click on **Build Now**.



Jenkins Job menu

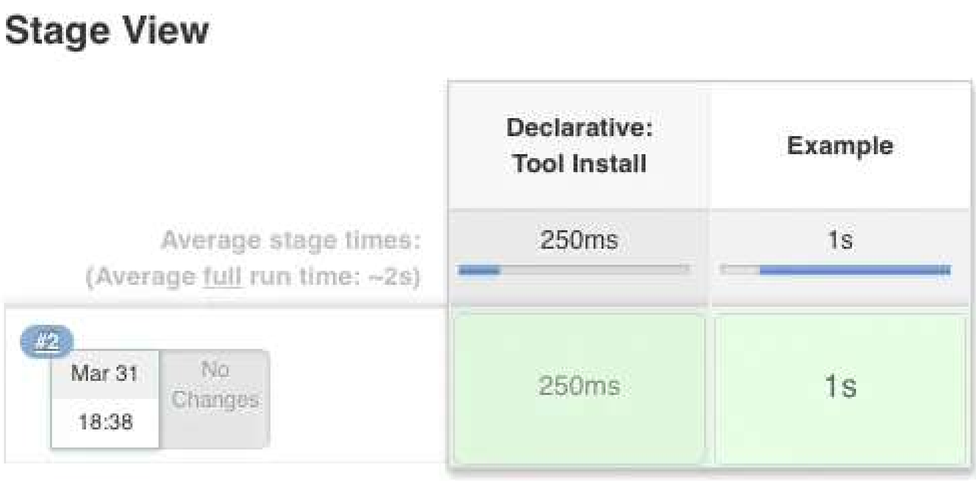
Wait some time for the build apear in **build history** below the menu (If necessary, reload the page).



Build Histoy

After the build over, the ball can be red or blue. If the ball is blue, the job is builded with success.

In center of page, you can visualize the stage view.



```
Started by user Gustavo
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/build-test
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Tool Install)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Example)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
[build-test] Running shell script
+ npm config ls
; cli configs
metrics-registry = "https://registry.npmjs.org/"
scope = ""
user-agent = "npm/5.6.0 node/v9.10.1 linux x64"
; node bin location =
/var/jenkins_home/tools/jenkins.plugins.nodejs.tools.NodeJSInstallat
ion/node/bin/node
; cwd = /var/jenkins_home/workspace/build-test
; HOME = /var/jenkins_home
; "npm config ls -l" to show all defaults.
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

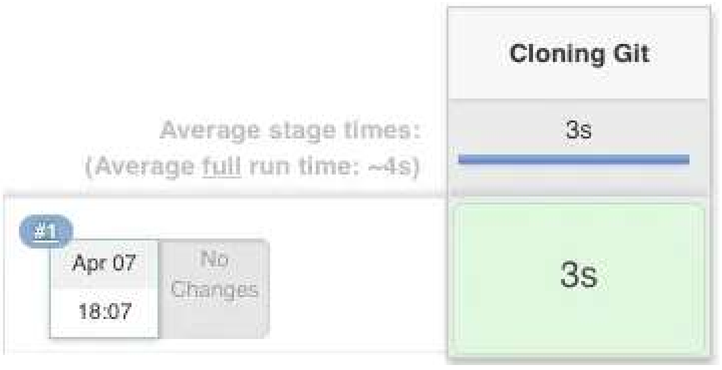
Ok. The build test right. Let's create a real node job.

Testing the application

Go to jenkins's home, click on "new job". Select pipeline and create the job named **"node_test"**, and put this script:

```
pipeline {
  agent any

  stages {
    stage('Cloning Git') {
      steps {
        git 'https://github.com/gustavoapolinario/node-todo-
frontend'
      }
    }
  }
}
```

Now, in job screen. Click on “configure” and let’s change the code.



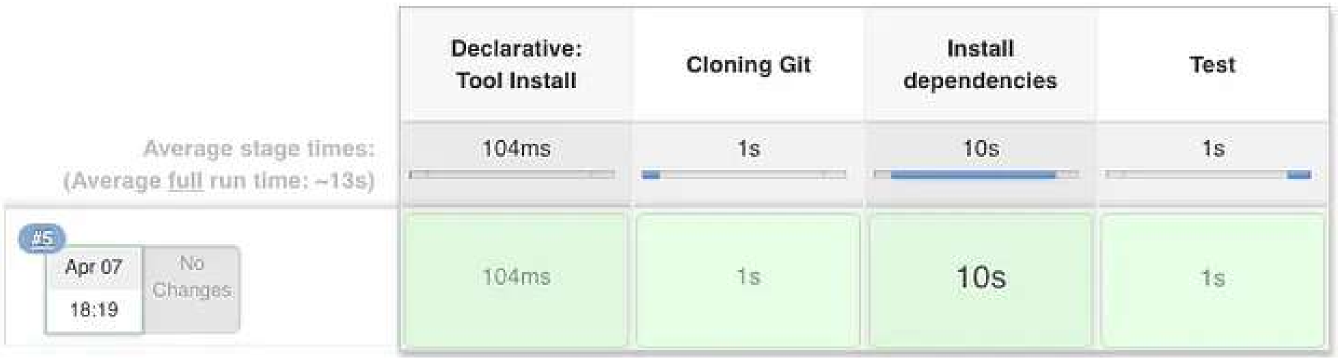
Stage View Git clone

Now, in job screen. click configure and let’s change the code.

```
pipeline {
  agent any

  tools {nodejs "node"}

  stages {

    stage('Cloning Git') {
      steps {
        git 'https://github.com/gustavoapolinario/node-todo-
frontend'
      }
    }

    stage('Install dependencies') {
      steps {
        sh 'npm install'
      }
    }

    stage('Test') {
      steps {
         sh 'npm test'
```

```
            }
        }
    }
}
```

After cloning step, we install the npm dependencies. Finally, the test is executed.



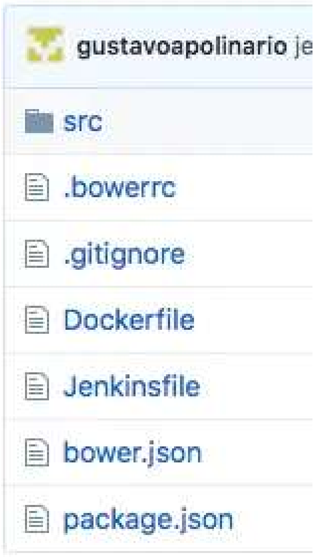
	Declarative: Tool Install	Cloning Git	Install dependencies	Test
Average stage times: (Average full run time: ~13s)	104ms	1s	10s	1s
#5 Apr 07 18:19 No Changes	104ms	1s	10s	1s

Stage View with git, download dependneices and npm test

All right, the application is tested with success.

Pipeline inside project

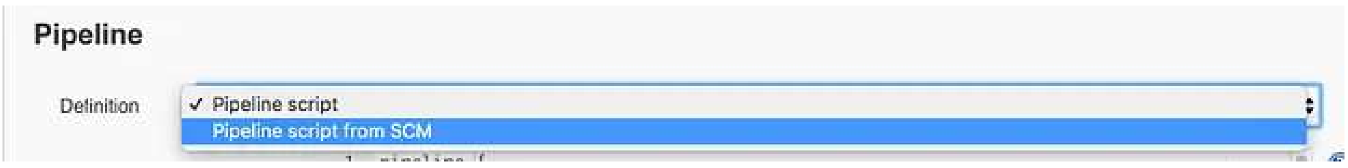
Now, put the script of pipeline inside your git project in a file named Jenkinsfile.



Jenkinsfile inside git project

Go back to jenkins and change the job.

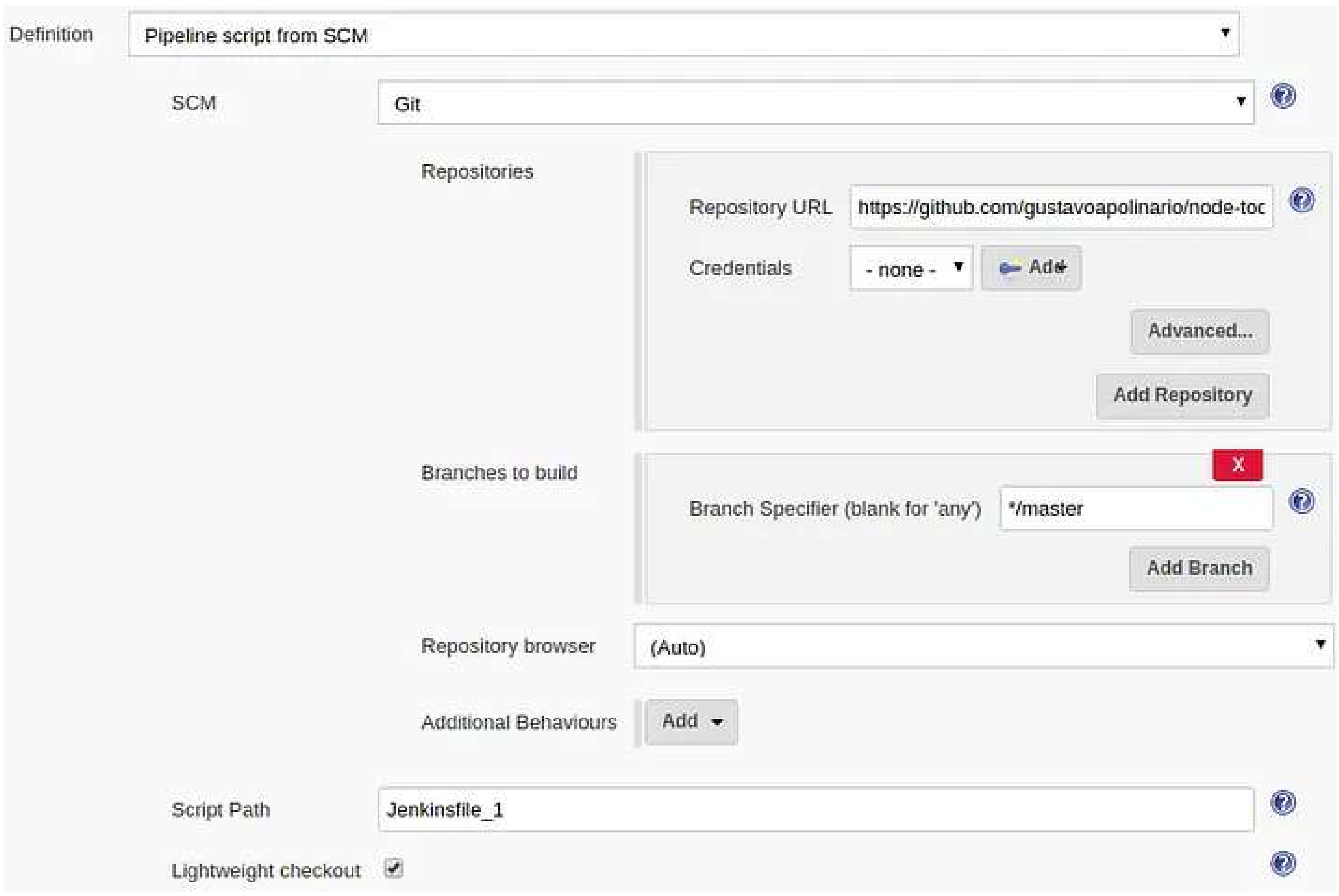
First step, change the definition of pipeline job to “**Pipeline script from SCM**”.



Pipeline script from SCM

Select **Git** on SCM Select.

Populate the field **Repository URL with:** "https://github.com/gustavoapolinario/node-todo-frontend" and put in **Script Path** field: "Jenkinsfile_1"



Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/gustavoapolinario/node-toc

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

Additional Behaviours

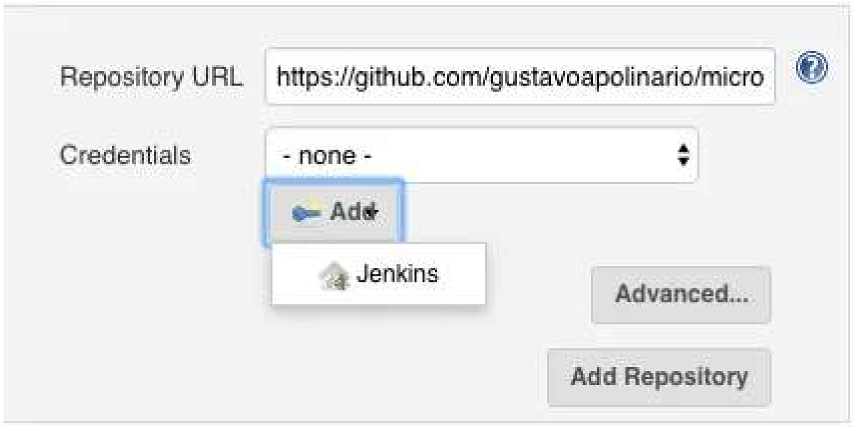
Script Path: Jenkinsfile_1

Lightweight checkout: ✓

Build your pipeline from git repository

Git credentials

If you need credential to access your git repository, click on "Add" button.

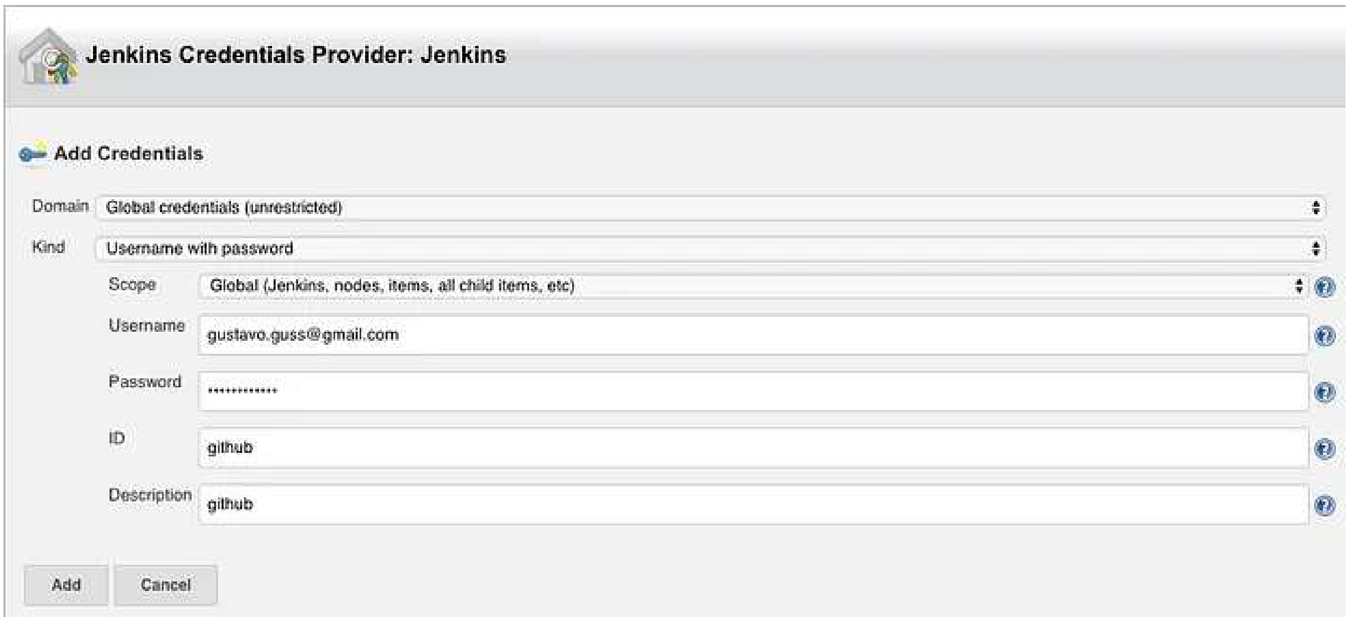


Repository URL: https://github.com/gustavoapolinario/micro

Credentials: - none -

Button Add git Credentials

Add your git credentials

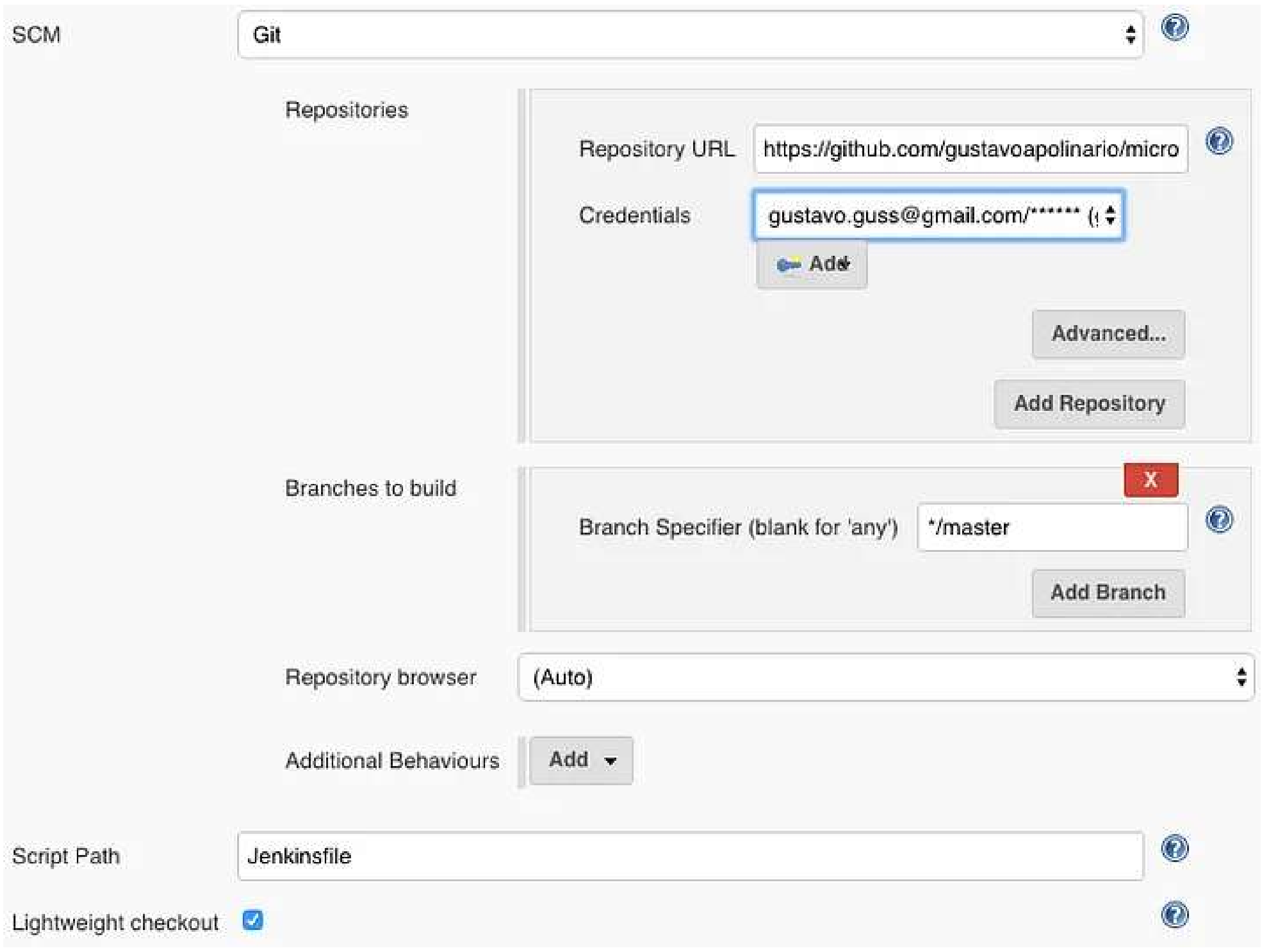

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain	Global credentials (unrestricted)
Kind	Username with password
Scope	Global (Jenkins, nodes, items, all child items, etc)
Username	gustavo.guss@gmail.com
Password	************
ID	github
Description	github

Add Credentials to git

And the configuration will be right with your credentials.



SCM	Git
Repositories	
Repository URL	https://github.com/gustavoapolinario/micro
Credentials	gustavo.guss@gmail.com/****** (
Branches to build	
Branch Specifier (blank for 'any')	*/master
Repository browser	(Auto)
Additional Behaviours	
Script Path	Jenkinsfile
Lightweight checkout	✓

We have the node project tested. Now we can build the docker container with our project.

More

To continue, you can learn how to build docker image, send to registry after test your node.js pipeline. See the next tutorial: Jenkins Building Docker Image and