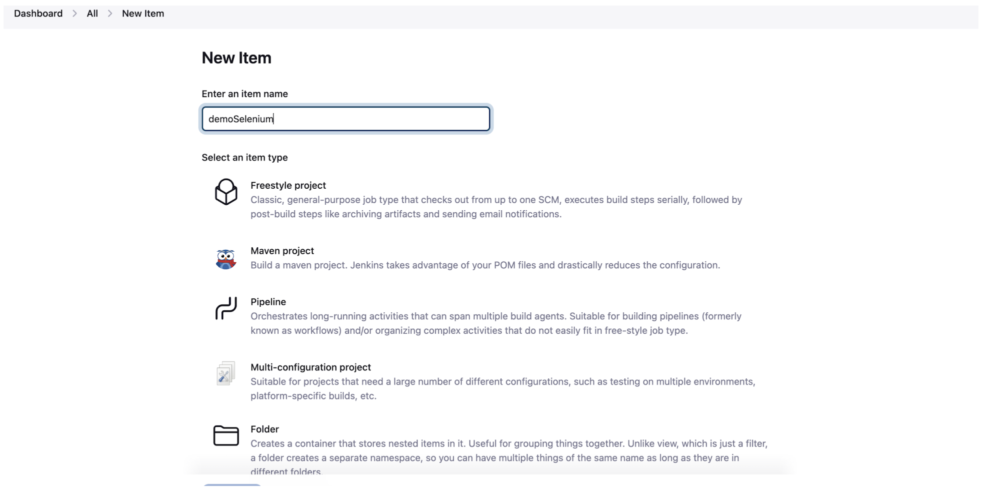
Step 1: Create a new item, select maven.


Dashboard > All > New Item

New Item

Enter an item name

demoSelenium

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

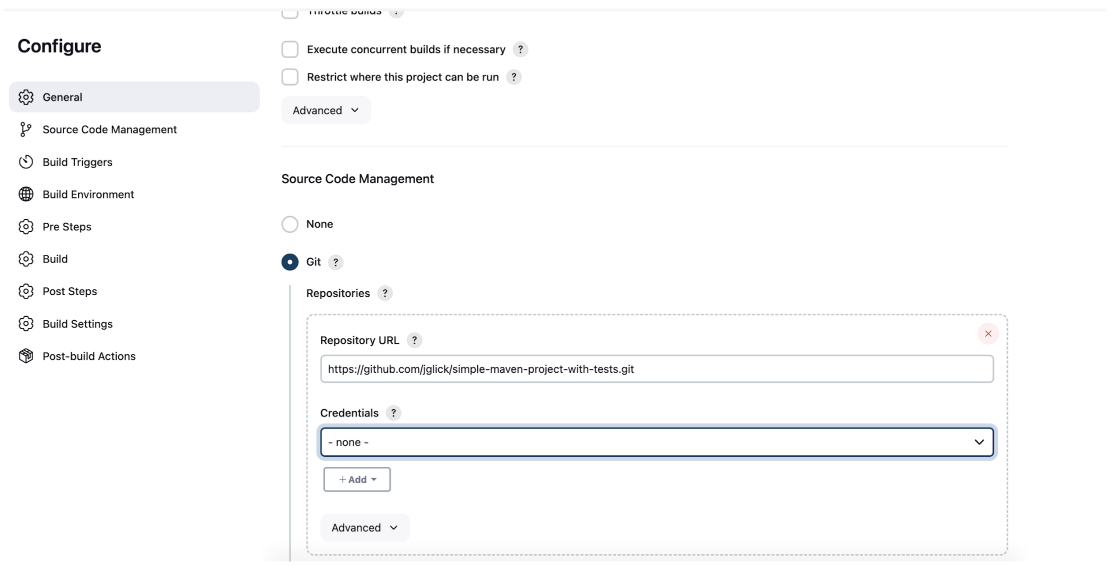
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Step 2: Paste the git repository url.



Execute concurrent builds if necessary

Restrict where this project can be run

Source Code Management

None

Git [✓]

Repositories

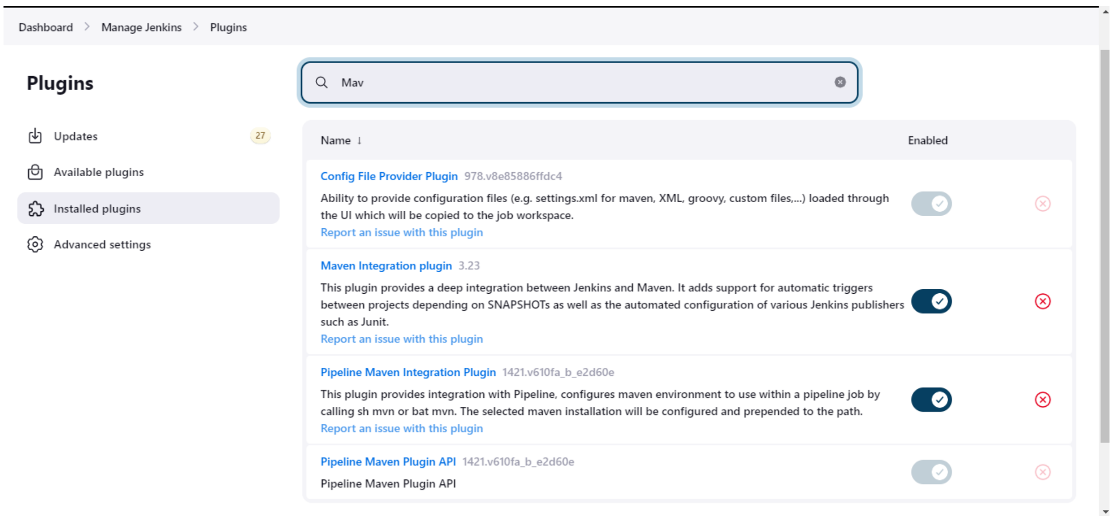
Repository URL

https://github.com/jglick/simple-maven-project-with-tests.git

Credentials

- none -

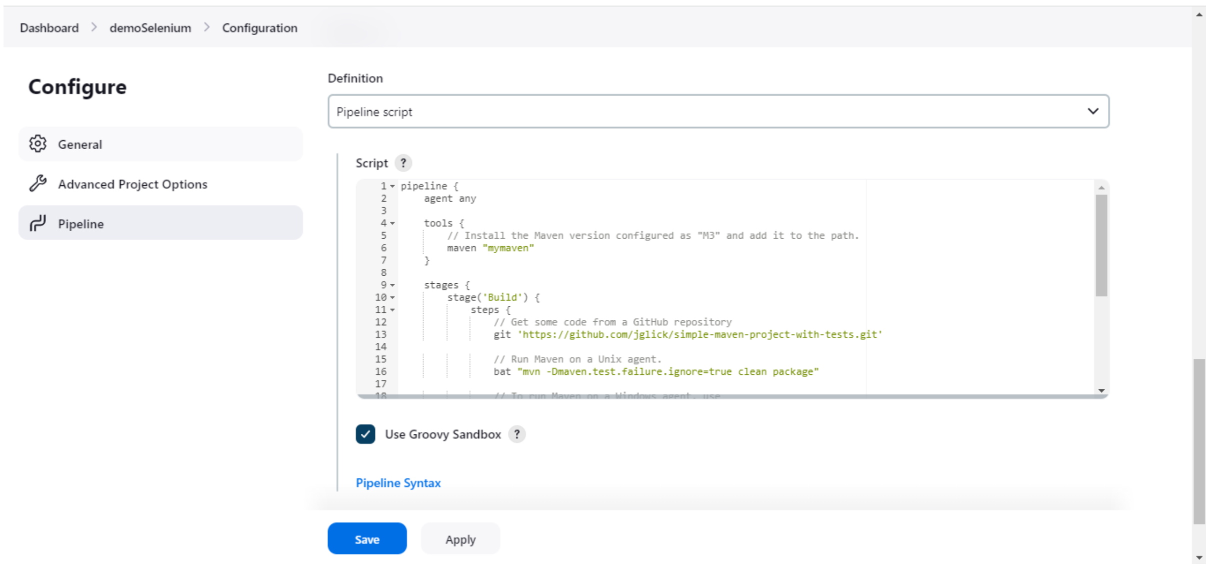
Step 3: Maven plugin.


Dashboard > Manage Jenkins > Plugins

Mav

Name ↓	Enabled
Config File Provider Plugin 978.v8e85886ffdc4 Ability to provide configuration files (e.g. settings.xml for maven, XML, groovy, custom files,...) loaded through the UI which will be copied to the job workspace. Report an issue with this plugin	✓
Maven Integration plugin 3.23 This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit. Report an issue with this plugin	✓
Pipeline Maven Integration Plugin 1421.v610fa_b_e2d60e This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path. Report an issue with this plugin	✓
Pipeline Maven Plugin API 1421.v610fa_b_e2d60e Pipeline Maven Plugin API	✓

Step 4: Pipeline script.



Definition: Pipeline script

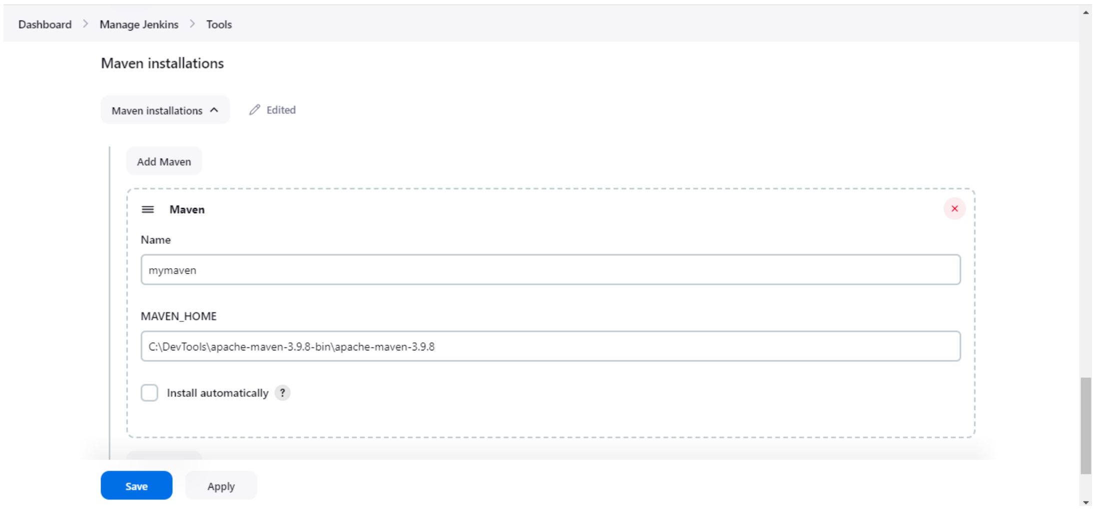
Script

```
pipeline {
    agent any

    tools {
        // Install the Maven version configured as "M3" and add it to the path.
        maven "mymaven"
    }

    stages {
        stage('Build') {
            steps {
                // Get some code from a GitHub repository
                git 'https://github.com/jglick/simple-maven-project-with-tests.git'

                // Run Maven on a Unix agent.
                bat "mvn -Dmaven.test.failure.ignore=true clean package"

                // To run Maven on a Windows agent, use
```

[✓] Use Groovy Sandbox

Pipeline Syntax

Step 5: Maven installation.

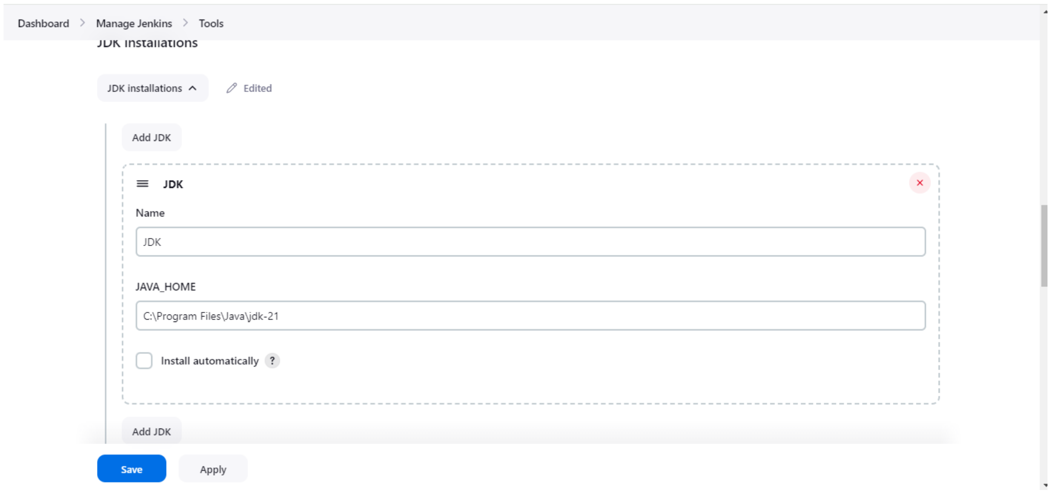


Name: mymaven

MAVEN_HOME: C:\DevTools\apache-maven-3.9.8-bin\apache-maven-3.9.8

[ ] Install automatically

Step 6: JDK installation.

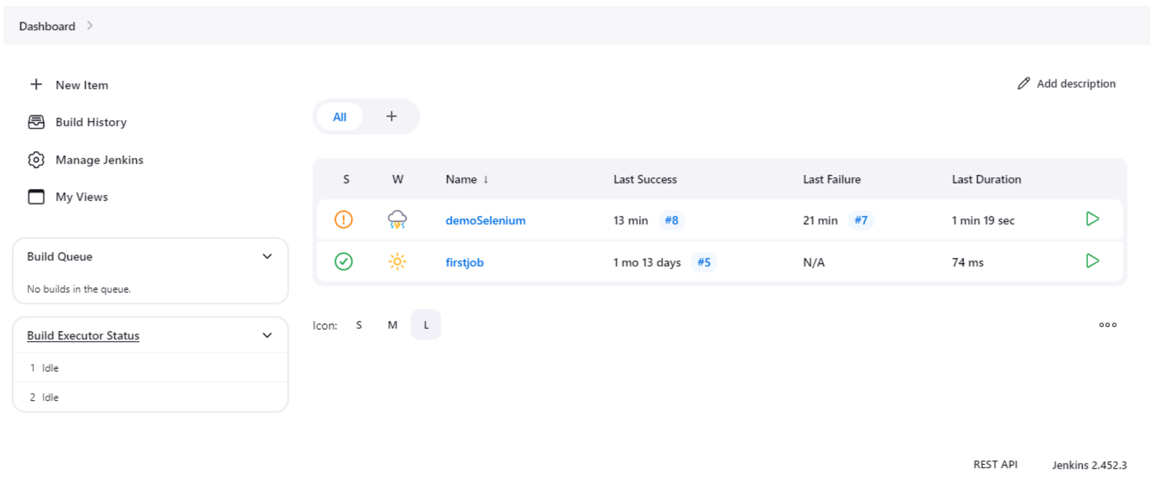


Name: JDK

JAVA_HOME: C:\Program Files\Java\jdk-21

[ ] Install automatically

Step 7: Build now.



Dashboard >

+ New Item
Build History
Manage Jenkins
My Views

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle

Add description

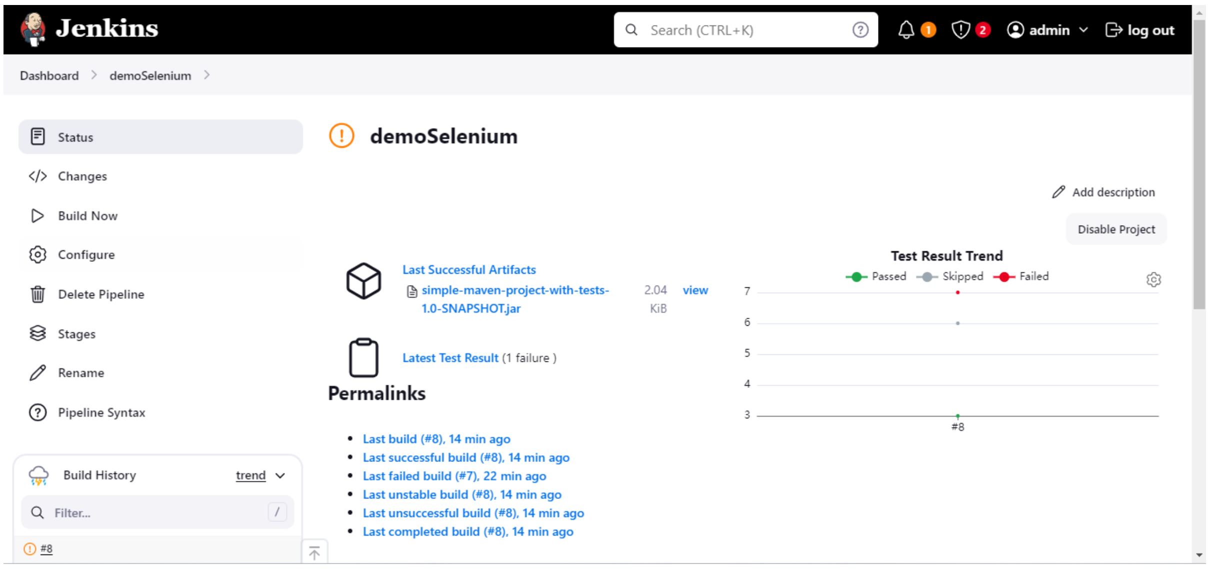
All +

S	W	Name ↓	Last Success	Last Failure	Last Duration	
!		demoSelenium	13 min #8	21 min #7	1 min 19 sec	▷
✓		firstjob	1 mo 13 days #5	N/A	74 ms	▷

Icon: S M L

REST API Jenkins 2.452.3

Step 8: See status.



Jenkins Search (CTRL+K) 1 2 admin log out

Dashboard > demoSelenium >

Status
Changes
Build Now
Configure
Delete Pipeline
Stages
Rename
Pipeline Syntax

Build History trend
Filter...
#8

demoSelenium

Add description
Disable Project

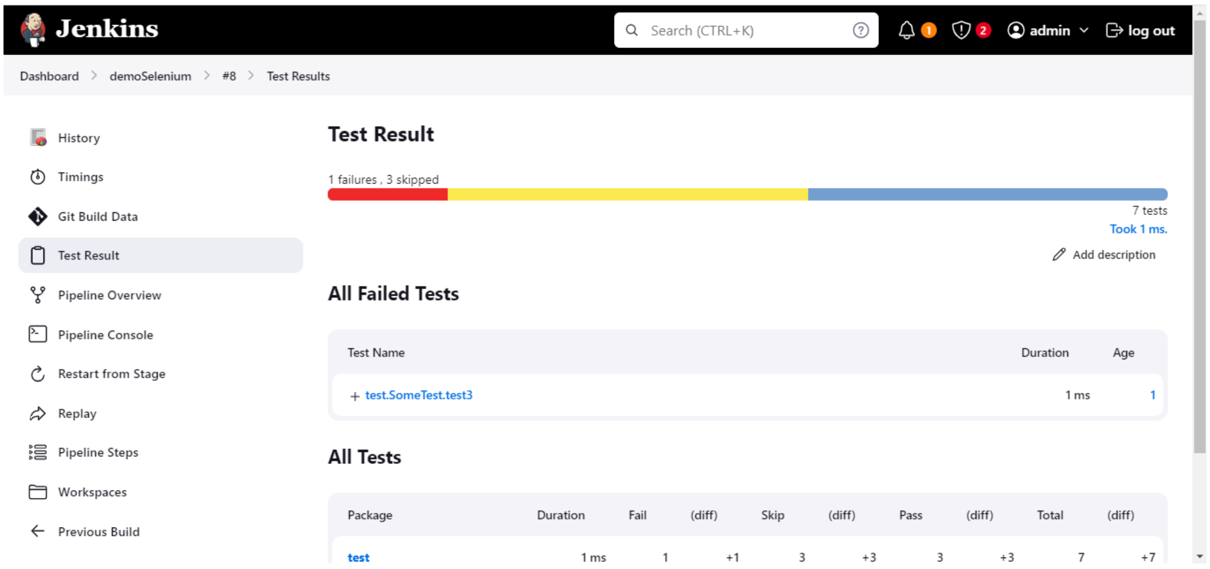
Last Successful Artifacts
simple-maven-project-with-tests-1.0-SNAPSHOT.jar 2.04 KiB view

Latest Test Result (1 failure)

Permalinks

- Last build (#8), 14 min ago
- Last successful build (#8), 14 min ago
- Last failed build (#7), 22 min ago
- Last unstable build (#8), 14 min ago
- Last unsuccessful build (#8), 14 min ago
- Last completed build (#8), 14 min ago

Step 9: Test result.



Jenkins Search (CTRL+K) 1 2 admin log out

Dashboard > demoSelenium > #8 > Test Results

History
Timings
Git Build Data
Test Result
Pipeline Overview
Pipeline Console
Restart from Stage
Replay
Pipeline Steps
Workspaces
Previous Build

Test Result

1 failures , 3 skipped

7 tests
Took 1 ms.

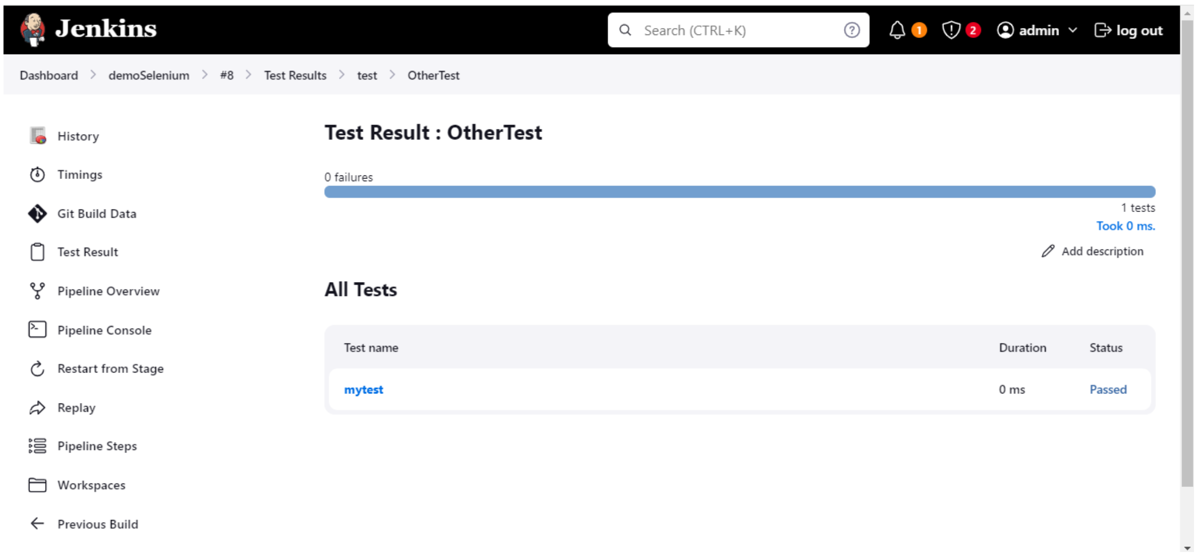
Add description

All Failed Tests

Test Name	Duration	Age
+ test.SomeTest.test3	1 ms	1

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Pass	(diff)	Total	(diff)
test	1 ms	1	+1	3	+3	3	+3	7	+7



Jenkins · Search (CTRL+K) · admin · log out

Dashboard > demoSelenium > #8 > Test Results > test > OtherTest

- History
- Timings
- Git Build Data
- Test Result
- Pipeline Overview
- Pipeline Console
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces
- Previous Build

Test Result : OtherTest

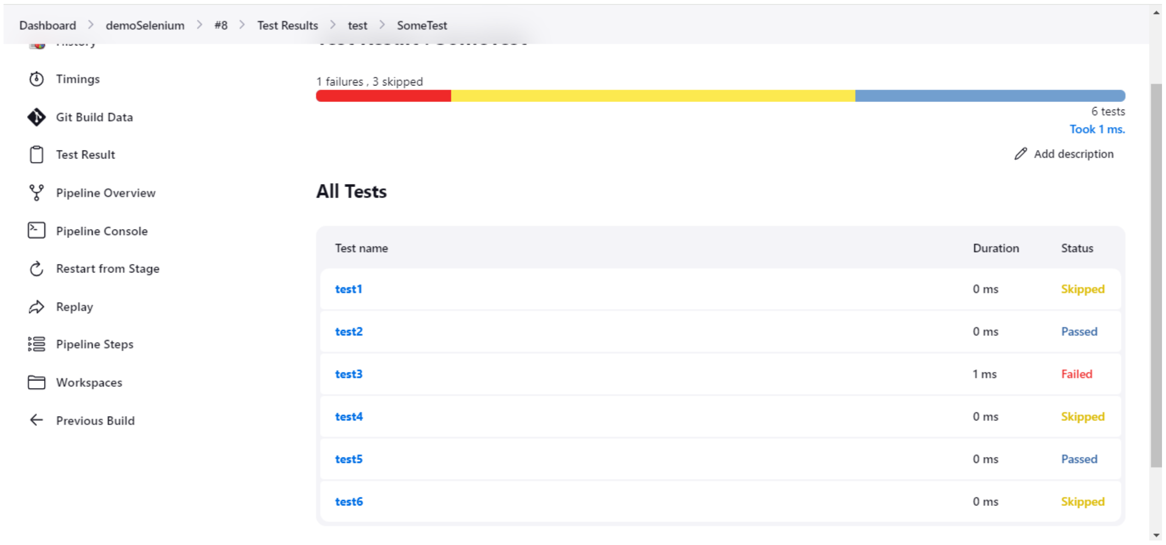
0 failures

1 tests
Took 0 ms.

Add description

All Tests

Test name	Duration	Status
mytest	0 ms	Passed



Dashboard > demoSelenium > #8 > Test Results > test > SomeTest

- Timings
- Git Build Data
- Test Result
- Pipeline Overview
- Pipeline Console
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces
- Previous Build

1 failures , 3 skipped

6 tests
Took 1 ms.

Add description

All Tests

Test name	Duration	Status
test1	0 ms	Skipped
test2	0 ms	Passed
test3	1 ms	Failed
test4	0 ms	Skipped
test5	0 ms	Passed
test6	0 ms	Skipped

Step 11: Code that is executed.

```java
package test;

import static org.junit.Assert.*;
import org.junit.internal.AssumptionViolatedException;

class Base {

    protected void run() {
        double r = Math.random();
        if (r < 0.1) {
            fail("oops");
        } else if (r < 0.2) {
            throw new AssumptionViolatedException("skipping");
        }
    }

}
```

Step 10: pom.xml

4.0.0 test simple-maven-project-with-tests 1.0-SNAPSHOT jar MIT License http://opensource.org/licenses/MIT org.apache.maven.plugins maven-surefire-plugin 2.18.1 junit junit 4.13.1 test UTF-8 1.8 1.8

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>test</groupId>
<artifactId>simple-maven-project-with-tests</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>jar</packaging>
<licenses>
<license>
<name>MIT License</name>
<url>http://opensource.org/licenses/MIT</url>
...
</license>
...
</licenses>
<build>
<plugins>
<plugin>
<groupId>org.apache.maven.plugins</groupId>
<artifactId>maven-surefire-plugin</artifactId>
<version>2.18.1</version>
...
</plugin>
...
</plugins>
...
</build>
<dependencies>
<dependency>
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>4.13.1</version>
<scope>test</scope>
...
</dependency>
...
</dependencies>
<properties>
<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
<maven.compiler.source>1.8</maven.compiler.source>
<maven.compiler.target>1.8</maven.compiler.target>
...
</properties>
```