

Help Document:

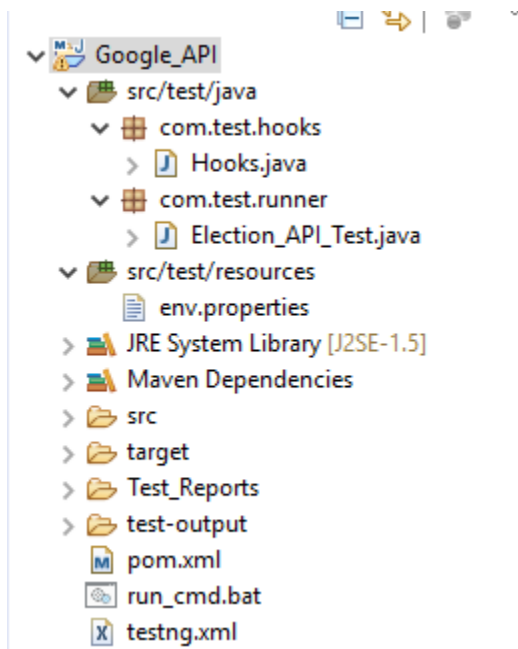
API Automation Testing

This document explained everything that we used in this framework and automated scripts.

Preconditions:

Before you import and run this script, JDK (Java) should be installed in your machine and Maven, TestNG plugins in your eclipse then only you'll get successful import project or else you'll see issues.

Packages and classes information:

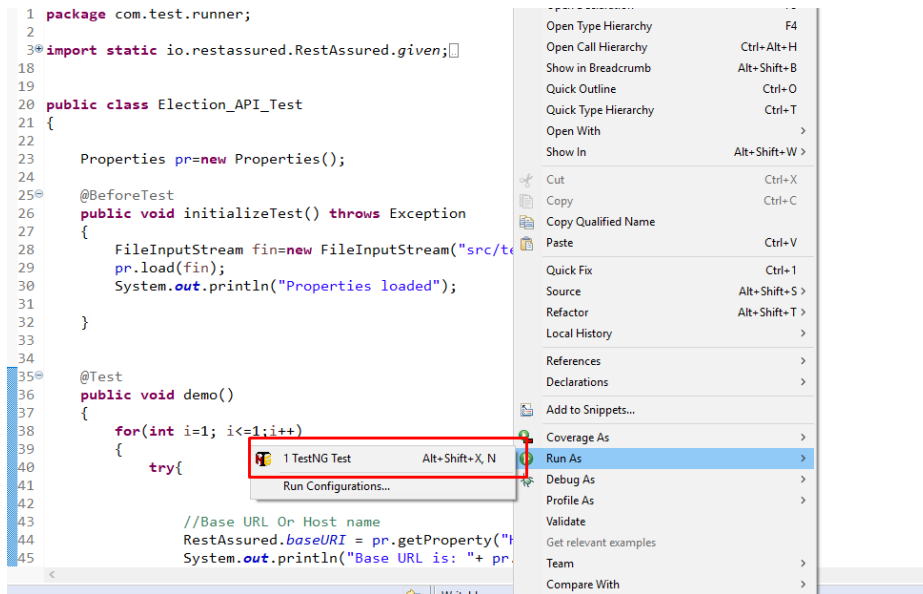


Package1: com.test.hooks - This package contain only one class called **Hooks**

Hooks: It is a reusable class where we defined few methods and reporting initialization as well as few global variables.

Package2: com.test.runner- This package contains only one class called “**Election API Test**”.

Election API Test: This class will be the runner class and it has code for automating API, you can run this file directly using TestNG.

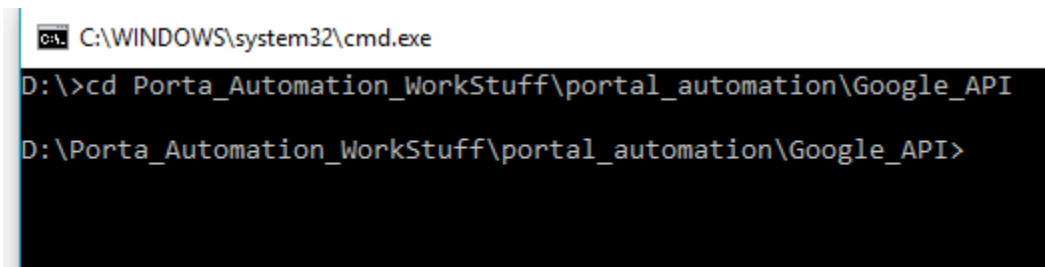


Package3: Under the resources we have one file called **env.properties** and this file useful to declare data example base URL and other based on requirement.

How to run script from command prompt:

Use below steps:

1. Open command prompt
2. Navigate to project repository location in cmd



3. Enter cmd called "***mvn -test***"

```
C:\WINDOWS\system32\cmd.exe - mvn test
```

```
D:\>cd Porta_Automation_WorkStuff\portal_automation\Google_API
```

```
D:\Porta_Automation_WorkStuff\portal_automation\Google_API>mvn test
```

```
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for google:Google_
API:jar:0.0.1-SNAPSHOT
[WARNING] 'dependencies.dependency.(groupId:artifactId:type:classfier)' must be unique: org.t
estng:testng:jar -> version 6.14.2 vs 6.9.10 @ line 63, column 13
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability o
f your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malform
ed projects.
```

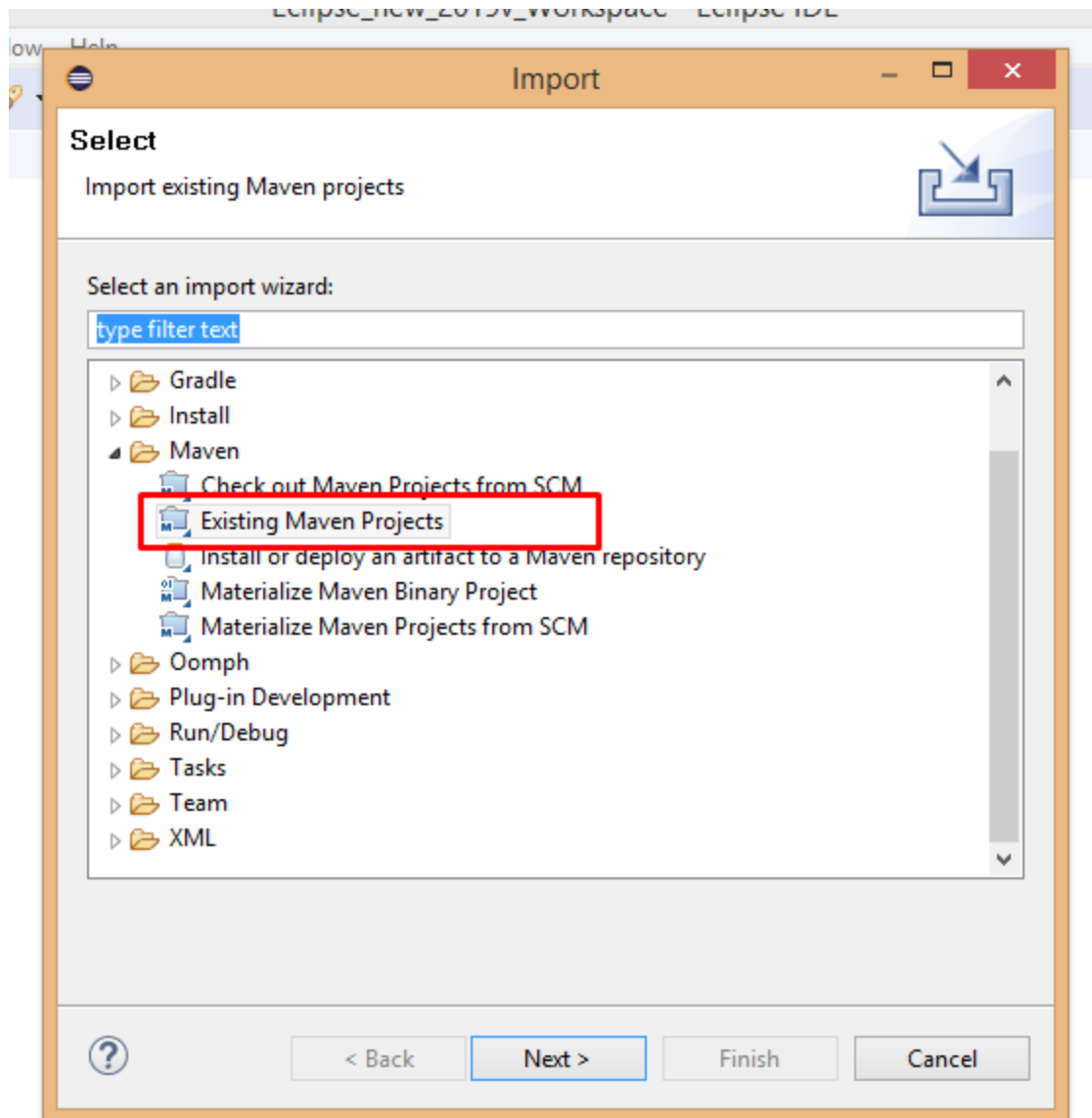
```
-----
T E S T S
-----
Running com.test.runner.Election_API_Test
Configuring TestNG with: org.apache.maven.surefire.testng.conf.TestNG652
Properties loaded
Base URL is: https://www.googleapis.com
```

After the `MVN -Test` command you'll be able to see something like above and you'll see the `TESTS` option like above then it will execute program as well.

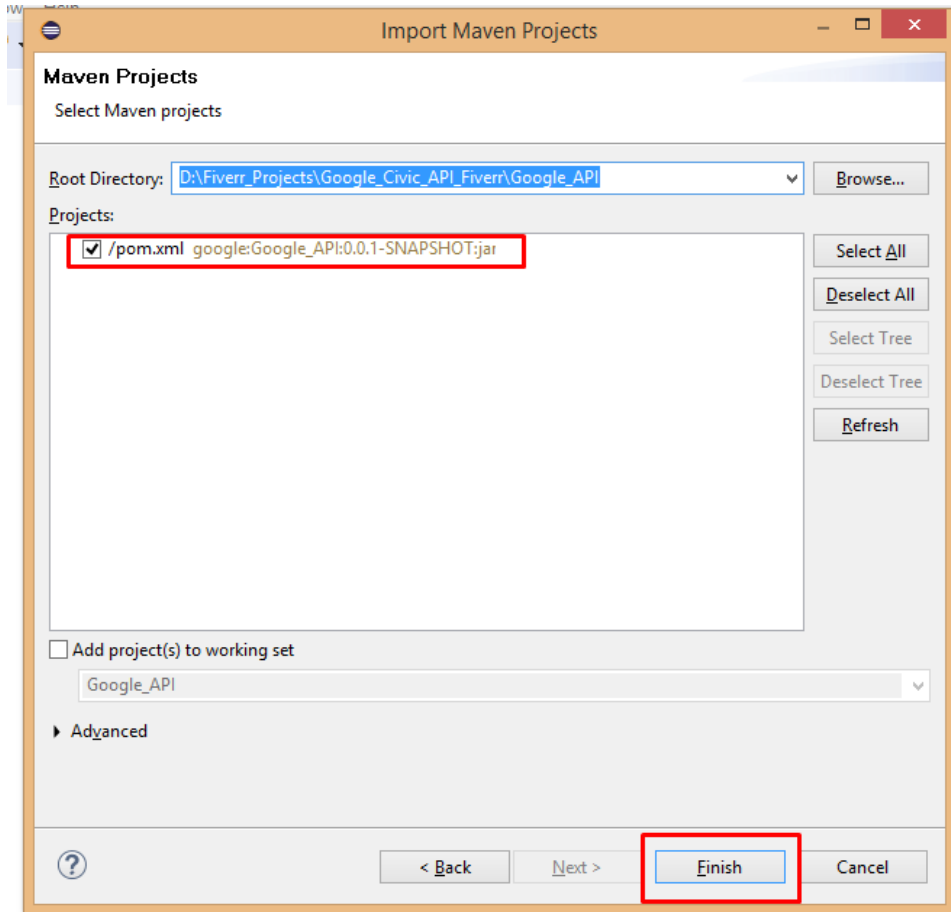
How to Import this project in Eclipse:

Please find below steps

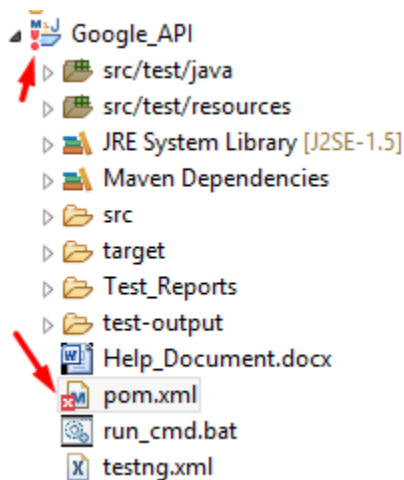
1. Choose file
2. Import
3. Existing Maven project under the Maven



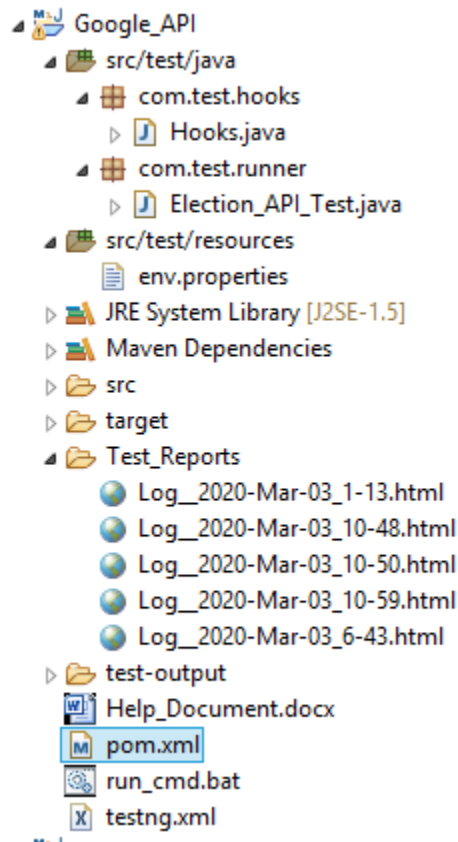
4. Browse file and observe result and click on Ok.



5. Make sure you wait till project is loaded successfully without any errors.
6. You may get this below symbol/error after importing if any dependency is incompatible with your environment, we should update the version number and save the project it will be sort.



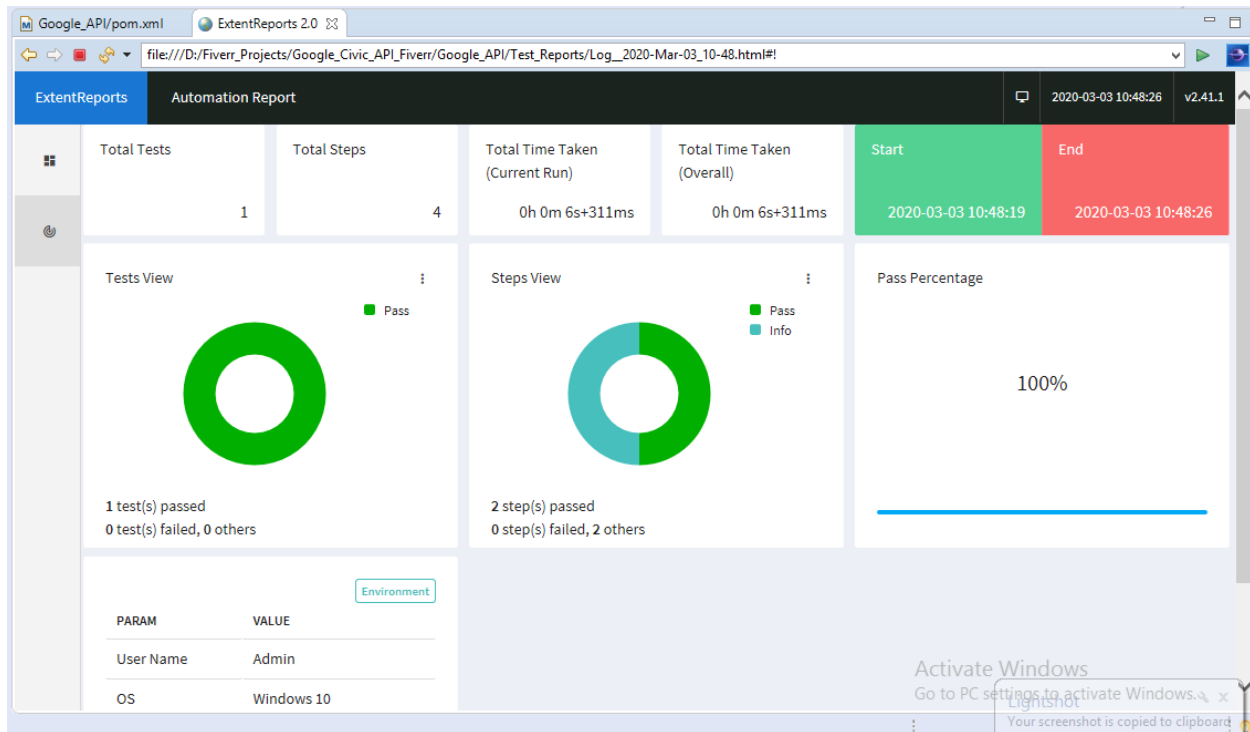
7. After the successful import you should be able to see the below structure without any error marks.



8. **Note:** Open *run_cmd.bat* file and change the location path as per your system path.

Test Output Look like below: Need to refresh this Test_Reports folder after the execution.

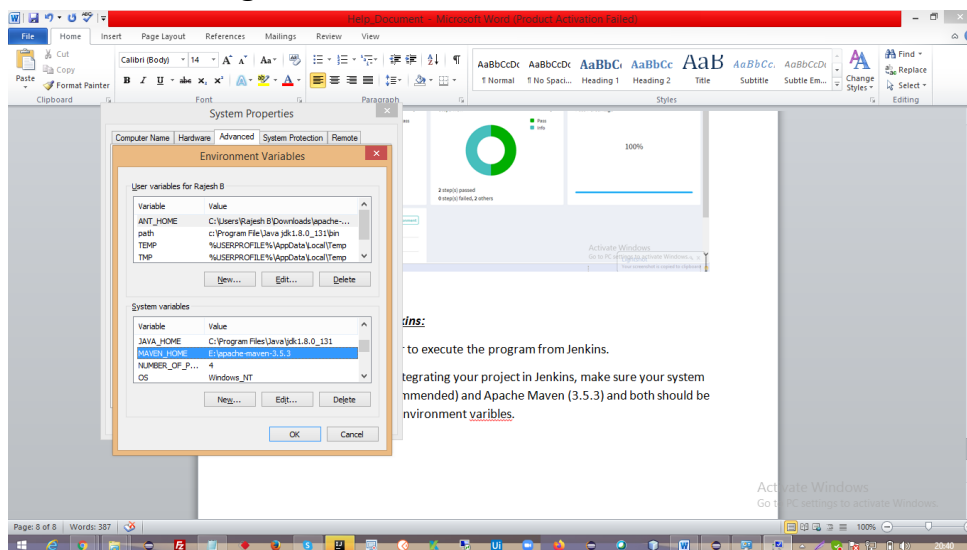
After the execution test out may look like below screenshots with clear information.



How to Setup test in Jenkins:

Find below steps in-order to execute the program from Jenkins.

1. Before you start Integrating your project in Jenkins, make sure your system install JDK (jdk8v recommended) and Apache Maven (3.5.3) and both should be configured under Environment variables.



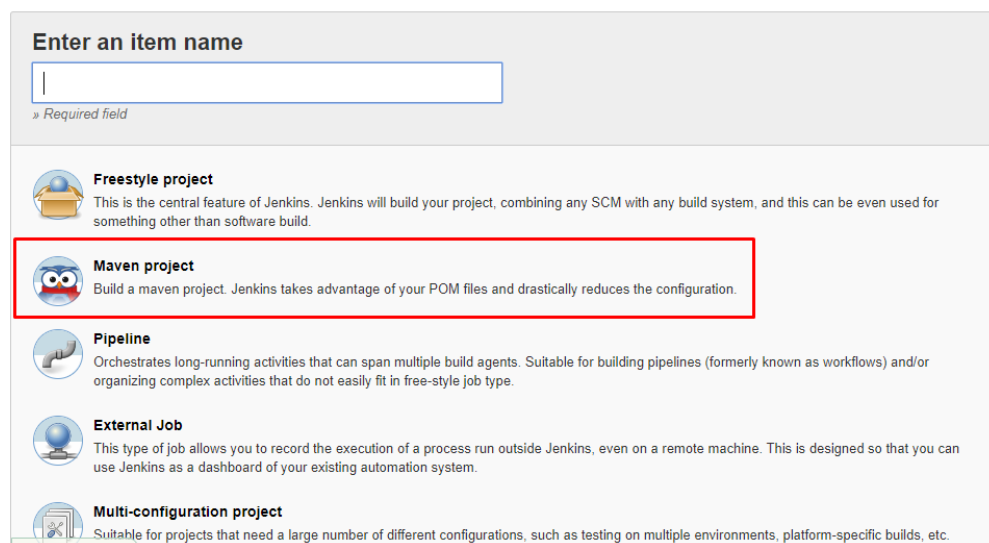
2. Open pom.xml and add maven surefire plugin with profile name as below.

```
<profiles>
  <profile>
    <id>run</id>
    <build>
      <pluginManagement>
        <plugins>
          <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>2.21.0</version>
            <configuration>
              <suiteXmlFiles>
                <suiteXmlFile>testng.xml</suiteXmlFile>
              </suiteXmlFiles>
            </configuration>
          </plugin>
        </plugins>
      </pluginManagement>
    </build>
  </profile>
</profiles>
```

So if you can observe, we given profile name as **run** this we should configure in Jenkins in order to activate our script.

Steps from Jenkins side:

1. Enter project name and choose Maven



The screenshot shows the Jenkins 'Enter an item name' dialog. At the top, there is a text input field for the item name, with a 'Required field' label below it. Below the input field, there is a list of project types with their respective icons and descriptions:

- 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.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration. (This option is highlighted with a red border in the image.)
- 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.
- External Job**: This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

2. Please Enter your description

General Source Code Management Build Triggers Build Environment Pre Steps Build Post Steps Build Settings

Post-build Actions

Description **The google API**

[Plain text] [Preview](#)

- ☐ Discard old builds
- ☐ GitHub project
- ☐ This project is parameterized
- ☐ Throttle builds
- ☐ Disable this project
- ☐ Execute concurrent builds if necessary

JDK **jdk1.8.0_131**

JDK to be used for this project

[Advanced...](#)

Activate Wir

3. Please Choose/Enter as below

Build

Root POM **pom.xml**

Goals and options **clean compile test**

MAVEN_OPTS

4. Choose custom workspace

☒ Use custom workspace

Directory **D:\Fiverr_Projects\Google_Civic_API_Fiverr\Google_API**

Maven Validation Level **DEFAULT**

Settings file **Use default maven settings**

Global Settings file **Use default maven global settings**

5. Under Build Triggers please enter the schedule time

The screenshot shows the 'Build Triggers' configuration panel. The title 'Build Triggers' is highlighted with a red box. Below the title, there are several checkboxes: 'Build whenever a SNAPSHOT dependency is built' (checked), 'Schedule build when some upstream has no successful builds' (unchecked), 'Trigger builds remotely (e.g., from scripts)' (unchecked), 'Build after other projects are built' (unchecked), and 'Build periodically' (checked, also highlighted with a red box). Below the 'Build periodically' checkbox, there is a 'Schedule' label and a text input field containing the cron expression '49 22 * * *'. A green circular icon with a 'G' is visible in the bottom right corner of the input field. Below the input field, there is a warning message: '⚠ Spread load evenly by using 'H 22 * * *' rather than '49 22 * * *'. Would last have run at Monday, 2 March, 2020 10:49:53 PM IST; would next run at Tuesday, 3 March, 2020 10:49:53 PM IST.'

6. Under the post build actions choose TestNG Results

The screenshot shows the 'Post-build Actions' configuration panel. The title 'Post-build Actions' is at the top. Below it, there is a list of actions. The first action, 'Publish TestNG Results', is highlighted with a red box. To the right of this action is a red 'X' icon. Below the action name, there is a label 'TestNG XML report pattern' and a text input field containing the pattern '**/testng-results.xml'. To the right of the input field is a blue question mark icon. Below the input field, there is a button labeled 'Advanced...'. At the bottom of the panel, there is a button labeled 'Add post-build action' with a dropdown arrow.

7. Save the project

8. You'll be able to see below kind of reports after the execution

```
"name": "Rhode Island Providence Ward 1 Special Election",
"electionDay": "2020-04-07",
"ocdDivisionId": "ocd-division/country:us/state:ri"
},
{
  "id": "4981",
  "name": "Rhode Island Providence Ward 1 Special Primary Election",
  "electionDay": "2020-03-03",
  "ocdDivisionId": "ocd-division/country:us/state:ri"
}
]
}
```

Test Passed: Both Actual and Expected response is same.

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 10.24 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[JENKINS] Recording test results

[WARNING] Attempt to (de-)serialize anonymous class hudson.maven.reporters.SurefireArchiver\$2; see: <https://jenkins.io/redirect/serialization-of-anonymous-classes/>

[INFO] -----

[INFO] BUILD SUCCESS

[INFO] -----

[INFO] Total time: 37.391 s

[INFO] Finished at: 2020-03-03T22:51:01+05:30

[INFO] -----

Waiting for Jenkins to finish collecting data

Activate Windows
Go to PC settings to activate

And other type of results too.