using maven "test" command to run testcases

App.java

(main class/service class)

AppTest.java

(Test class)

//App.java

package com.nt.test;

public class App

package com.nt.test;

{

public int sum(int a,int b) { return a+b;

}

public static void main(String[] args)

{

import static org.junit.Assert.assertEquals; import static org.junit.Assert.assertTrue;

import org.junit.Test;

*** Unit test for simple App. */**

System.out.println("Hello World!"); System.out.println("1234");

/**

} }

public class AppTest

{

@Test

public void testSumWithPositivies()

=>TestCase is a test plan where the expected results will be matched with actual results (Each @Test method represents one TestCase)

=> Maven Project gives JUnit Dependency as the default dependency which that So we can write the TestCases will be executed automatically in in "test" phase of the maven life cycle

default

=> EVery TestCase is a Test Plan in the form of @Test method in TestCase class

=> In TestCase class, the testcase methods will be there annotated with @Test

=> To practice this code

a) create maven project using

maven-archetype-quickstart

b) change the java version from

the pom.xml (1.7 to 17) and update the Project

{

int val1=10;

```
}
int val2=20;
int expected=30;
App app=new App();
int actual-app.sum(val1, val2);
assertEquals(expected, actual);
```

Each testcase is a testplan

where the expected results

will be matched with actual results

```
@Test
public void testSumWithNegetives()
{
int val1=-10;
}
}
}
int val2=-20;
int expected=-31;
App app=new App();
int actual app.sum(val1, val2);
assertEquals(expected, actual);
@Test
public void testSumZeros()
{
int val1=0;
int val2=0;
int expected=0;
App app=new App();
int actual-app.sum(val1, val2);
assertEquals(expected, actual);
```

testcase1

testcase2

testcase3

Right click on the project ---> run as ---> maven build.. ---> goals :: test

```
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
```

[INFO] BUILD SUCCESS

[INFO]

[INFO] Total time: 1.992 s

[INFO] Finished at: 2023-09-04T09:28:13+05:30 [INFO]

Generating "test report" of junit using maven surefire-report plugin

=> This plugin generates the html file as the report having detailed information about the junit test cases that are executed

right click on the project ---> run as ---> maven build... --> goals :: surefire-report:report

--> refresh the project ---> expand target folder

plugin name goal

->apply -->ok

✓ target

> generated-sources

> generated-test-sources

>maven-status

Every Plugin contains multiple goals representing multiple tasks that is can perform. surefire-report plugin goals are

The Surefire Report Plugin has three goals:

- surefire-report:report Generates the test results report into HTML format.

⚫ surefire-report:report-only This goal does not run the tests, it only builds the report. ...

•

surefire-report:failsafe-report-only This goal does not run the tests, it only builds the IT reports.

Built by:

maven

Surefire Report

Summary

[Summary] [Package List] [Test Cases]

Tests

Errors

3

0

✓ site

> css

>

Failures

0

images

surefire-report.html

open in the browser

Skipped

0

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

executing maven "site" command/plugin to get complete information about the project

right click on the Project ---> run as ---> maven build ... ----> goals :: site refresh the project ---> expand the target folder

✓ target

✓ site

> css

> images

dependencies.html

dependency-info.html

index.html

plugin-management.html

plugins.html

project-info.html

summary.html

M pom.xml

open with web browser

MavenProj01

Last Published: 2023-09-04 | Version: 0.0.1-SNAPSHOT

Project Documentation

▾ Project Information

Dependencies

Dependency

Information

About

Plugin Management

Plugins Summary

Project Summary

Project Information

Built by:

maven

Field

Name

Description

Homepage

Project Organization

This project does not belong to an organization.

Value

MavenProj01

http://www.example.com

Executing java App using maven command/plugin

=======

note:: The eclipse maven standalone app can be executed using using eclipse options

a) run as ---> java App

Here we are executing maven Project

b) use run (play button) of the menubar c) ctrl+f11 or f11

Success Rate 100%

what is the difference b/w surefire-report:report and surefire-report:report-only goals of the surefire-report plugin?

Ans) "report" goal of the surefire-report plugin not only executes junit test code but also generates the test report having surefire.html name (target/reports folder) "report-only" goal of the surefire-report plugin does not execute junit test code but generates the test report having surefire.html file name (target/reports folder)

F3: To open the source code of certain code that is present in the eclipse Editor ctrl+shift+T :: To open the source of certain java class by searching for it

as normal Eclipse IDE application

in

note:: There is not phase called "run" or "execute" in maven default Life cycle.. So we need to configure additional plugin called "maven-exec-puglin" as shown specifying the main class name

if want to execute through maven, then we need special plugin pom.xml file

a) add plugin in pom.xml under <plugins> tag

```
<plugin>
<groupId>org.codehaus.mojo</groupId>
<artifactId>exec-maven-plugin</artifactId>
<version>1.6.0</version>
<configuration>
<mainClass>com.nt.test.App</mainClass>
</configuration>
</plugin>
```

b) run the app using maven build

right click --> run as ---> build .. ---> goals :: exec:java

Why there is no "run" or "execute" plugin in maven built-in Default Life cycle?

Ans) In real projects, we do not run the App/Code for testing, we actually run the Junit test cases for testing the code, In that process code runs indirectly.. So maven has given "test" phase .. not the run /execute phase

=>creating, deploying, executing the java web application using maven

ste 1) create maven project using "maven-archetype-webapp"

File menu ----> new ---> maven project ---> next ---> select "maven-archetype-webapp" ----> next --->

note:: only in maven-archetype-quickstart based projects the given default package will be created in other archetype projects that will not created

step2)

add servlet-api dependency to pom.xml file

groupId: nit

org.apache.maven.archetypes

artifact id:: Maven Proj02-webapp version ::... default

package :: com.nt.servlet (will not be created) --->finish.

maven-archetype-webapp

if this archetype is not coming in the list then we add it explicitly

Add Archetype

<!-- https://mvnrepository.com/artifact/jakarta.servlet/jakarta.servlet-api -->

<dependency>

<artifactId>jakarta.servlet-api</artifactId>

<groupId>jakarta.servlet</groupId>

<version>5.0.0</version>

now 6.x

<scope>provided</scope>

</dependency>

collect from the mvnrepository.com

by searching for Jakarta servlet api

Archetype Group Id: org.apache.maven.archetypes

1.5

Archetype Artifact Id: maven-archetype-webapp Archetype Version: Repository URL:

`1.4 1.5`

step3) change java version to

21

in pom.xml by performing maven update <maven.compiler.source> 2k/maven.compiler.source> <maven.compiler.target: 2/maven.compiler.target>

right click on the Project --- maven ---> update project --> ....

step4) bring src/main/java, src/test/java folders to the Project stucture right click on project --> build path---> configure build path --->

order and export tab -->select last two checkboxes

Source

Projects Libraries Order and Export Build class path order and exported entries: (Exported entries are contributed to dependent projects)

MavenProj02-WebApp/src/main/java

MavenProj02-WebApp/src/test/java

JRE System Library [JavaSE-17] Maven Dependencies

b

step3-extension) change the Project facet's we module version to 5.0/6.0

Right click on the Project --->properties ----> Project facets ---> web module ---> version 6.0

In maven web application there are standard 5 folders src/main/java ----> To place the source code of the app src/main/resources ----> to place the supporting files related to source code src/test/java ---------> To place the unit test code of the app src/test/resources ----> To place the supporting files related to unit test code src/main/webapp -----> To place public resources of the web application like html files, jsp files, images, audio files, vedio files and etc,..

apply ---->close

step5) configure tomcat server with eclipse IDE

window menu --->preferences ---> run time env ---> server ----> Name:

add ---->select

apache ---> tomcat 10.x --->

Apache Tomcat v10.0 Tomcat installation directory: D:\Tomcat10.x

now 11.05

11.05

JRE:

Workbench default JRE

Browse...

apache-tomcat-10.0.13 Download and Install...

-->apply --->ok

Installed JREs...

note:: also add tomcat server, in the "server" tab

Go to server tab ---> click on the link --> add server -->select apache tomcat 10.x (now 11.05)

step5) develop the code

In Maven web application

src/main/java --->for placing the source code

src/test/java ----> for placing unit test code

src/main/webapp -------> for plblic area code (html, js, css, image, audio, vedio and etc.. files) src/main/resources ----> supporting files related to source code development (like properites files) ----> supporting files related to unit test code (like properties files)

Code src/test/resources

=====

index.xml

=>The content kept in these folders will be moved to different directories of java web application's

deployment directory structure like

=> src/main/java packages will be moved to WEB-INF/classes folder

=> src/main/webapp code moves to web application's Root Directory => src/main/resources code moves to WEB-INF/classes folder

<h1> <a href="dateurl">Show Date and Time</a></h1>

DateServlet.java

import java.util.Date;

import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.HttpServlet; import jakarta.servlet.http.HttpServletRequest; import jakarta.servlet.http.HttpServletResponse;

@WebServlet("/dateurl")

public class DateServlet extends HttpServlet {

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

//get PrintWriter

PrintWriter pw=response.getWriter();

//set response content type

}

response.setContentType("text/html");

MavenProj02-WebApp

pw.println("<h1> date and time is ::"+new Date()+"</h1>");

23 Deployment Descriptor: <web app>

//add home hyperlink

Java Resources

src/main/java

pw.println("<br><a href='index.html'>home</a>");

//close the stream

pw.close();

protected void doPost(HttpServletRequest request, HttpServletRespo

doGet(request, response);

}

}

web.xml

<!DOCTYPE web-app PUBLIC

"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

"http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

</web-app>

step7) run the application

←

right click on the Project ---> run as ---> run on server ---> select Tomcat -> ....

→ C

localhost:6677/MavenWebApp/index.jsp

Show Season Name

preparing war file using maven

right click the Project ---> run as ---> maven build .. ---> goals ::

✓ target

> generated-sources

> m2e-wtp

>maven-archiver

> MavenProj02-WebApp

>maven-status

MavenProj02-WebApp.war

› za Deployment Descriptor: <web app>

com.nt.servlet

› za Deployment Descriptor: <web app>

›

DateServlet.java

src/test/java

23 Deployment Descriptor: <web app>

Libraries

> JRE System Library [JavaSE-17]

✓

Maven Dependencies

>junit-4.11.jar - C:\Users\NATARAJ\.m2\re

> hamcrest-core-1.3.jar - C:\Users\NATARA

OException {

>jakarta.servlet-api-5.0.0.jar - C:\Users\NA

> Deployed Resources

✓ src

main

> java

✓

webapp

WEB-INF

x web.xml

index.html

> test

› target

Mpom.xml

HTML to Servlet Communication

package

(or) goals :: clean package

To deploy this war file, copy the war file to <Tomcat_home>\webapps folder

Maven FAQs

What is Project build process?

What are different build tools?

What is difference b/w Ant build Tool and maven/gradle Build tool?

What is need of adding maven tool support on top of IDE based Project Development?

what are maven features?

What is POM.xml in maven Project?

What is archetype in maven and explain different archetypes?

Explain maven built-in life cycle and their phases ?

How can u say maven tool is self intelligent?

What is dependency management in maven ?

what is transitive dependency management in maven?

Explain various maven repositories and their relation?

Explain various goals that are there in maven build life cycle phases?

use

List out various scopes to use in dependency management?

What is the goal /phase we should to pack the maven project to war /jar file?

Can u explain group Id, artifactId, version w.r.t maven project/dependency /plugin?

What is the plugin we should use to generate Test Report?

What is the plugin we should use to generate whole project Report?

What is maven iheritence and how it is useful in real projects?

How to keep maven Project packed jar file in the Local Repository?

↑

home

↑

Q

localhost:6677/MavenWebApp/seasonurl

Summer Season