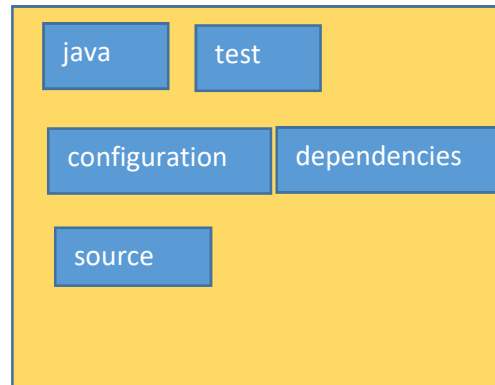
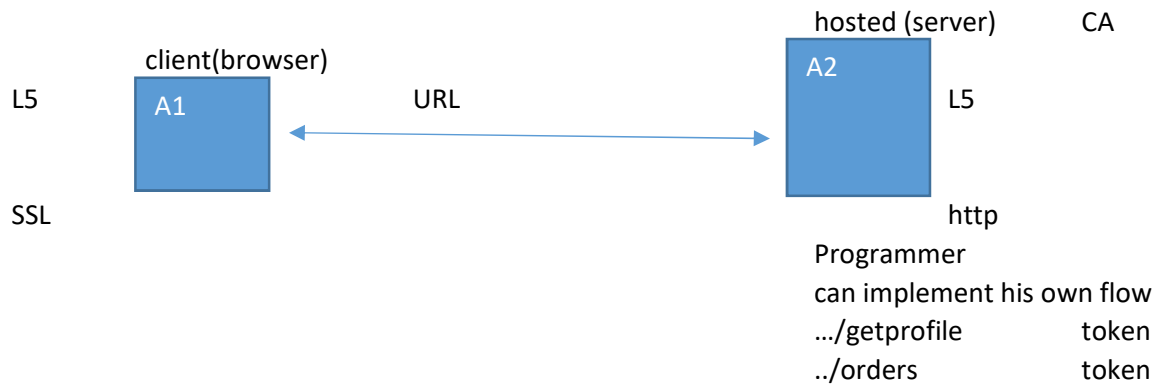


Enterprise level java application
different own structure
source
configurations
references
test cases
audio
video
image
properties

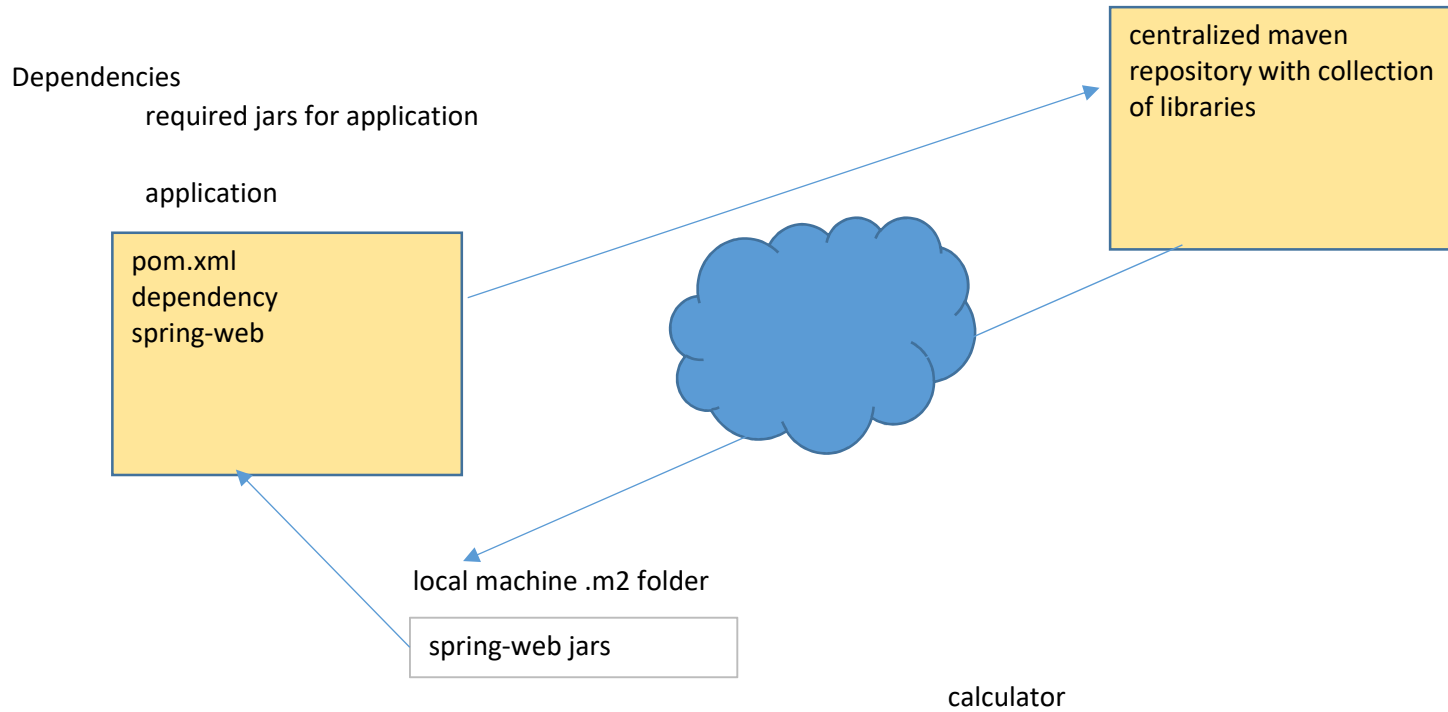


maven standalone quickstart
 web-application web

IDE



Secured sessions



Group ID
Defines package structure

Artifact ID
Defines project name

com.stackroute.app1.model
com.stackroute.app1.service
com.stackroute.app1.application

maven-quickstart
to create simple standalone (console) application with main()

maven-webapp
to create spring mvc based web application

maven quickstart application

New Project

Generators

- Maven Archetype
- JavaFX
- Kotlin Multiplatform
- Compose Multiplatform
- IDE Plugin
- Android

To create a general Maven project, go to the [New Project](#) page.

Name:

Location:

Project will be created in: C:\Wave-39-BE\Demos-Exers\Course1\Sprint1\Demo2\exer2

☐ Create Git repository

JDK: corretto-17 Amazon Corretto version 1

Catalog: Internal [Manage catalogs...](#)

Archetype:

Version:

Additional Properties

No properties

Advanced Settings

GroupId:

ArtifactId:

Version:

```
assertEquals();  
    expected actual  
assertEquals(16,square(4));
```

```
square(int x)  
..  
..
```

```
class MainClass
{
    Operations o1 = new Operations();
    sout(o1.add(4,5));    //9
}
```

```
class Operations{
    public int add(int a, int b) {
        return a+b;
    }
}
```

```
class TestOperations
{
```

```
    testcase1 add(2,3);    5
    testcase2 add(5,10);   15
```

```
}
```

get addition of two passed numbers (passing as parameters)

return type	int
method name	add
arg list	two int

```
int add(int a, int b) { }
```

Steps to create maven quick start application with test cases

Step 1 Create maven quick start project

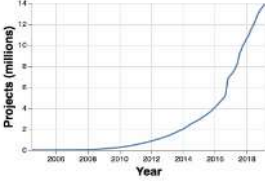
Step 2 add required dependency in pom.xml
 junit junit

← → ↻ 🔒 mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-

MVN REPOSITORY

Search for groups, artifacts, categories

Indexed Artifacts (31.2M)



Popular Categories

- Testing Frameworks
- Android Packages
- Logging Frameworks
- Java Specifications
- JSON Libraries
- Core Utilities
- JVM Languages
- Mocking
- Language Runtime
- Web Assets
- Annotation Libraries
- Logging Bridges
- HTTP Clients
- Dependency Injection
- XML Processing

Home » org.junit.jupiter » junit-jupiter

JUnit Jupiter API

JUnit Jupiter is the API for w

License	EPL 2.0
Categories	Testing Frameworks
Tags	junit testing
Ranking	#44 in MvnRe #3 in Testing
Used By	11,808 artifa

Central (58) ICM (2)

Ve

5.9.x	5.9.1
	5.9.0
	5.9.0-RC1
	5.9.0-M1
5.8.x	5.8.2
	5.8.1
	5.8.0
	5.8.0-RC1
	5.8.0-M1

search for junit
open the recent version
copy the dependency

[Home](#) » [org.junit.jupiter](#) » [junit-jupiter-api](#) » [5.9.1](#)



JUnit Jupiter API » 5.9.1

JUnit Jupiter is the API for writing tests using JUnit 5.

License	EPL 2.0
Categories	Testing Frameworks
Tags	junit testing api
HomePage	https://junit.org/junit5/
Date	Sep 20, 2022
Files	pom (3 KB) jar (202 KB) View All
Repositories	Central
Ranking	#44 in MvnRepository (See Top Artifacts) #3 in Testing Frameworks
Used By	11,808 artifacts

Maven

Gradle

Gradle (Short)

Gradle (Kotlin)

SBT

Ivy

Grape

Leiningen

```
<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.9.1</version>
  <scope>test</scope>
</dependency>
```

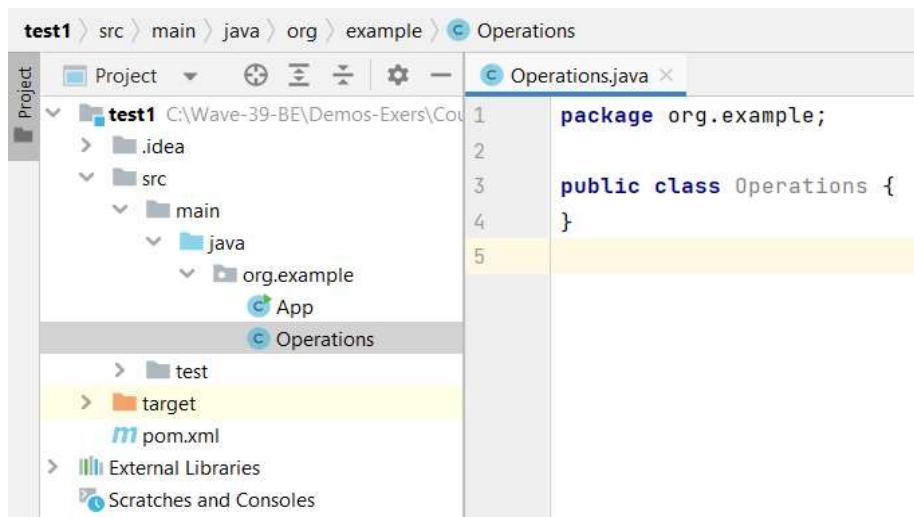
paste in local pom.xml
under <dependencies> element

```

19 <dependencies>
20
21 <!-- <dependency>-->
22 <!-- <groupId>junit</groupId>-->
23 <!-- <artifactId>junit</artifactId>-->
24 <!-- <version>3.8.1</version>-->
25 <!-- <scope>test</scope>-->
26 <!-- </dependency>-->
27
28 <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
29 <dependency>
30 <groupId>org.junit.jupiter</groupId>
31 <artifactId>junit-jupiter-api</artifactId>
32 <version>5.9.1</version>
33 <scope>test</scope>
34 </dependency>

```

Step 3 Define methods in some class

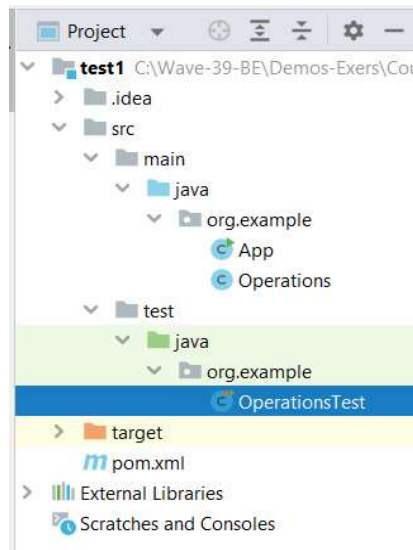
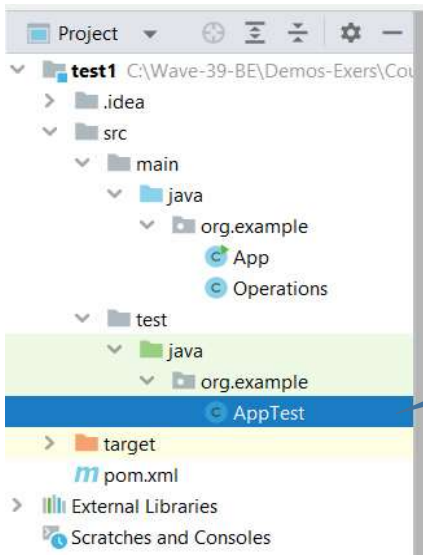


```

1 package org.example;
2
3 public class Operations {
4     public int add(int a, int b){
5         return a+b;
6     }
7     public int sub(int x, int y){
8         return x-y;
9     }
10 }

```

Step 4 Write testcase in testing file



```
7 import static org.junit.jupiter.api.Assertions.assertEquals;
8
9 public class OperationsTest {
10     3 usages
11     Operations op;
12     // before each test: allocate memory for object
13     @BeforeEach
14     public void setup(){
15         op = new Operations();
16     }
17     // after each test: de-allocate memory for object
18     @AfterEach
19     public void clean(){
20         op = null;
21     }
22     // testcase method(s)
23     @Test
24     public void testAdd1(){
25         // expecting 7 when add(3,4) is called
26         assertEquals( expected: 7, op.add( a: 3, b: 4));
27     }
28 }
```




```
4      public class Operations {
5          1 usage
6          public int add(int a, int b) { return a+b; }
7          public int sub(int x, int y) { return x-y; }
8          // get list of leap years between passed range of years
9          1 usage
10         public List<Integer> getLeaps(int from, int to){
11             List<Integer> leaps = new ArrayList<Integer>();
12             for(int y=from;y<=to;y++){
13                 if( y%400==0 || (y%100!=0 && y%4==0)) {
14                     leaps.add(y);
15                 }
16             }
17             System.out.println(leaps);
18             return leaps;
19         }
20         // get result as PASS/FAIL as per parameter marks
21         // each marks must be >=50 to PASS
22         3 usages
23         @ public String getResult(int... marks){
24             String result="PASS";
25             for(int m:marks){
26                 if(m<50){
27                     result="FAIL";
28                     break;
29                 }
30             }
31             return result;
32         }
33     }
34 }
```

```

2 usages
35 public int div(int a, int b){ // a/b
36     if(b==0) {
37         throw new ArithmeticException();
38     }
39     else{
40         return a/b;
41     }
42 }
43 // public --- getFib(int count){ 5
44 //     ..
45 //     ..
46 //     return { 0 1 1 2 3 }
47 //     return { 0 1 1 2 3 5 }
48 //
49 // }
50 }
51 // 1990 2010
52 // 1992 1996 2000 2004 2008

```

```

9 import static org.junit.jupiter.api.Assertions.assertEquals;
10 import static org.junit.jupiter.api.Assertions.assertThrows;
11
12 public class OperationsTest {
13     Operations op;
14     // before each test: allocate memory for object
15     @BeforeEach
16     public void setup() { op = new Operations(); }
17
18     // after each test: de-allocate memory for object
19     @AfterEach
20     public void clean() { op = null; }
21
22     // testcase method(s)
23
24     @Test
25     public void testAdd1(){
26         // expecting 7 when add(3,4) is called
27         assertEquals( expected: 7, op.add( a: 3, b: 4));
28
29

```

```

29
30  @Test
31  public void testGetLeaps1(){
32      //      assertEquals(1992,1996,2000,2004,2008, op.getLeaps(1990,2010);
33      assertEquals(Arrays.asList(1992,1996,2000,2004,2008), op.getLeaps(1990,2010));
34  }
35  @Test
36  public void testResult(){
37      assertEquals( expected: "PASS",op.getResult( ...marks: 67,87,98,90));
38      assertEquals( expected: "FAIL",op.getResult( ...marks: 47,87,98,90));
39  }
40  @Test
41  public void testResult2(){
42      assertEquals( expected: "FAIL",op.getResult( ...marks: 67,87,98,90,56,77,56,59,49));
43  }
44  }
45  @Test
46  public void testDivSuccess(){
47      assertEquals( expected: 3,op.div( a: 12, b: 4));
48  }
49  @Test
50  public void testDivFailure(){
51      assertThrows(ArithmeticException.class, ()->op.div( a: 12, b: 0 ));
52  }
53  // public void testFib(){
54  //      // assertEquals(0 1 1 2 3 5, getFib(6));
55  //      // assertEquals(0 1 1 , getFib(3));
56  //  }
57  }

```