



# Junit 5

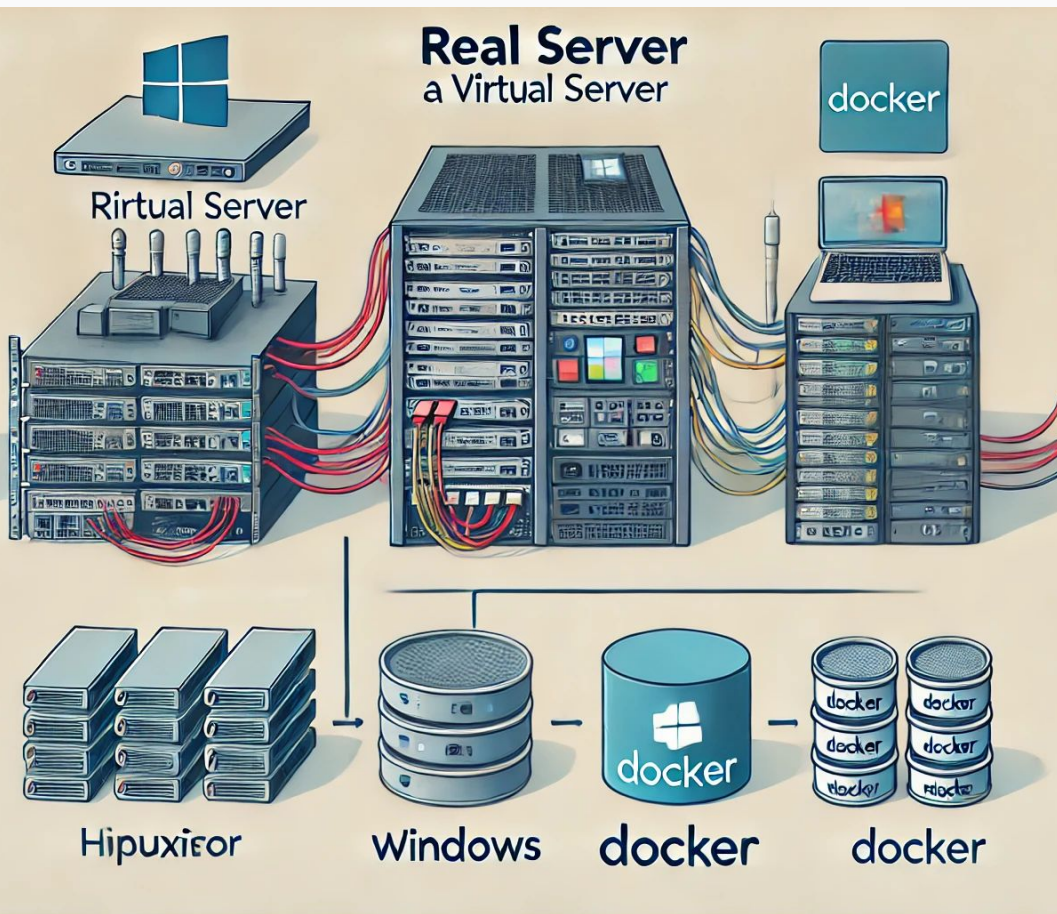
Execution

## We Cover

- Motivation
- Via IDE - IntelliJ
- Command line
- Surefire / Failsafe
- Parallel Execution
- Console Launcher

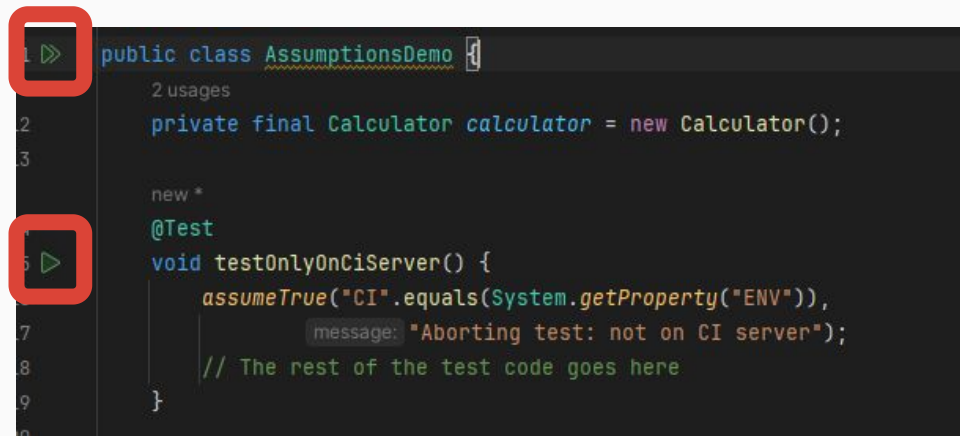


JUnit 5

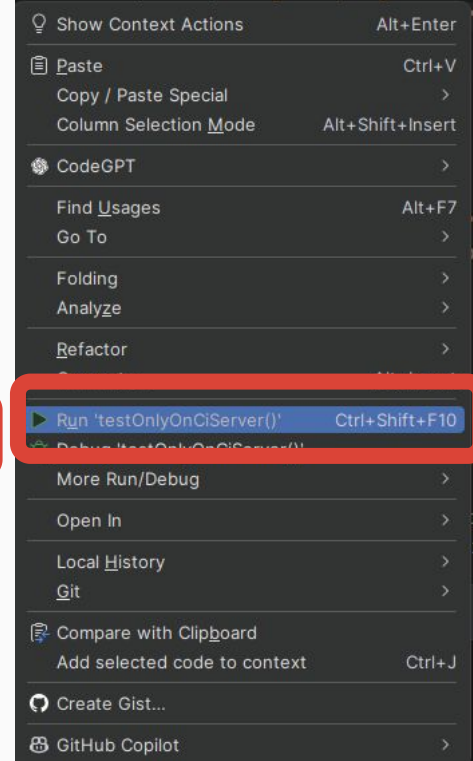
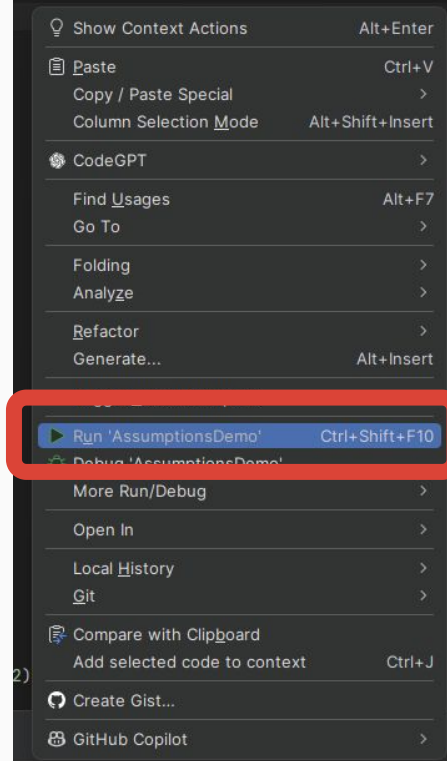


android





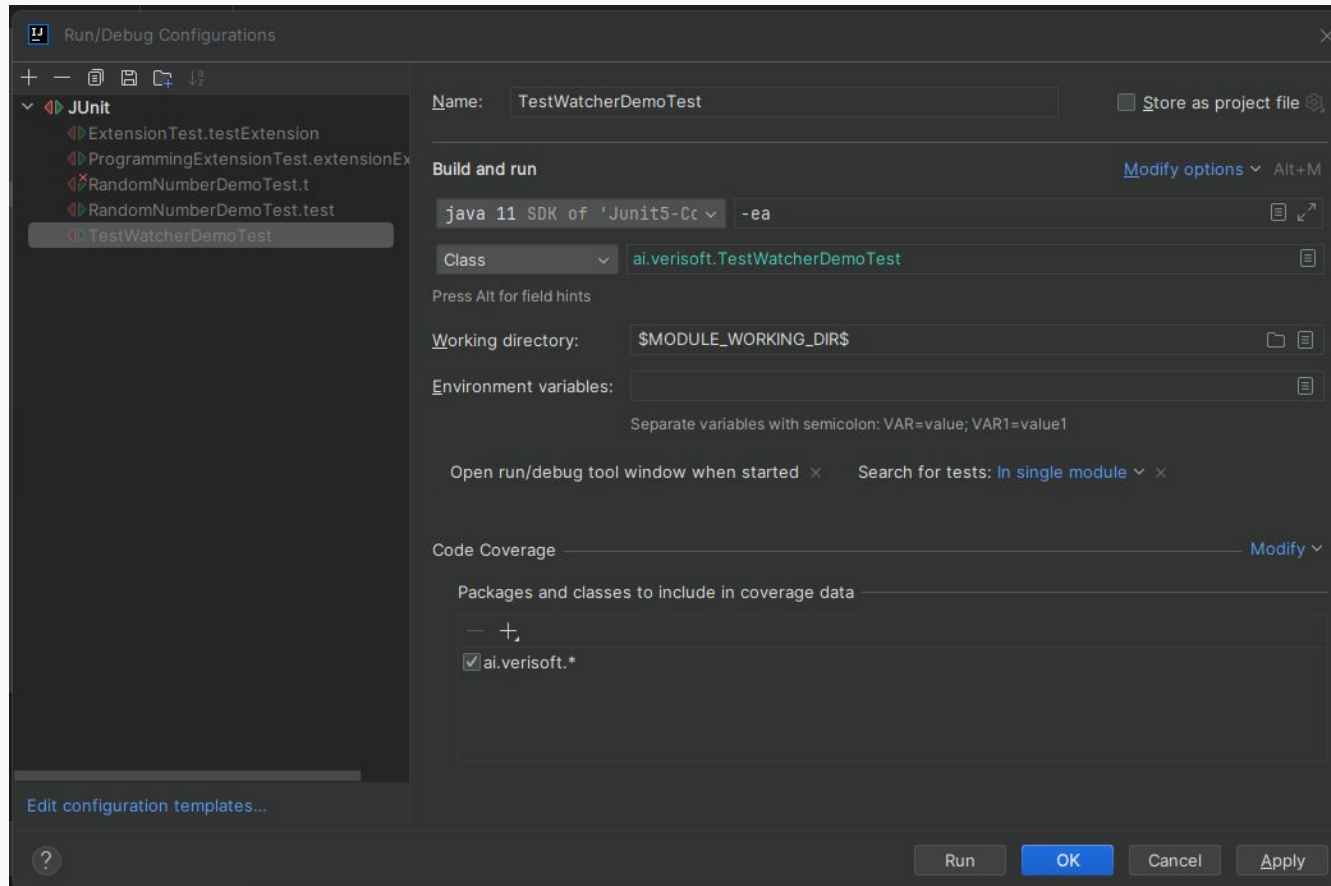
```
1 public class AssumptionsDemo {  
2     private final Calculator calculator = new Calculator();  
3  
4     new *  
5     @Test  
6     void testOnlyOnCiServer() {  
7         assumeTrue("CI".equals(System.getProperty("ENV")),  
8             message: "Aborting test: not on CI server");  
9         // The rest of the test code goes here  
10    }
```



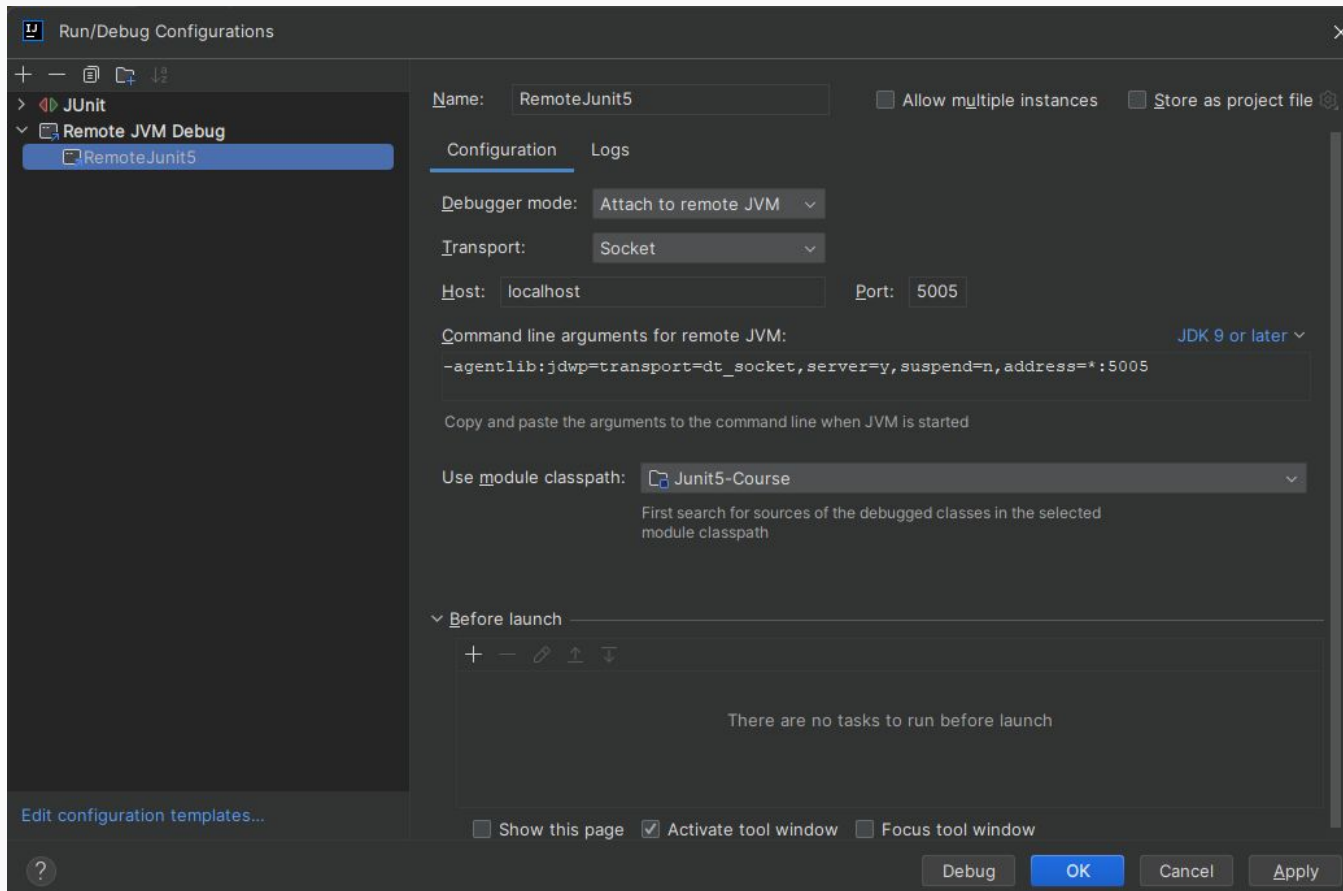
# Surefire basic parameters

Parameter	Description	Default Value
<code>-Dtest</code>	Specifies which test(s) to run (e.g., <code>-Dtest=MyTest</code> or <code>-Dtest=MyTest#myMethod</code> ).	Runs all tests
<code>-Dsurefire.includes</code>	Comma-separated list of test patterns to include (e.g., <code>**/MyTest.java</code> ).	<code>**/*.java</code>
<code>-Dsurefire.excludes</code>	Comma-separated list of test patterns to exclude.	None
<code>-Dsurefire.testFailureIgnore</code>	Ignores test failures (i.e., tests fail but the build succeeds).	<code>false</code>
<code>-Dsurefire.runOrder</code>	Specifies the order in which tests are run (e.g., <code>alphabetical</code> , <code>random</code> ).	<code>filesystem</code>
<code>-Dsurefire.reportFormat</code>	Specifies the format of the test report ( <code>brief</code> , <code>plain</code> , <code>xml</code> ).	<code>plain</code>
<code>-Dsurefire.failIfNoTests</code>	Fails the build if no tests are found.	<code>true</code>

# IntelliJ - Run / Debug Configuration



# IntelliJ - Debugging Using CLI



# Debugging from CLI

## Forked Process

```
mvn -Dmaven.surefire.debug test
```

```
mvn -Dmaven.surefire.debug="-agentlib:jdwp=transport=dt_socket,server=y,suspend=y,address=localhost:8000" test
```

## Non Forked Process

```
mvn -DforkCount=0 test
```

```
mvnDebug -DforkCount=0 test
```



# Surefire basic parameters

Parameter	Description	Default Value
<code>-Dtest</code>	Specifies which test(s) to run (e.g., <code>-Dtest=MyTest</code> or <code>-Dtest=MyTest#myMethod</code> ).	Runs all tests
<code>-Dgroups</code>	Specifies which test groups to include (e.g., <code>-Dgroups=integration</code> ).	Runs all groups
<code>-DexcludeGroups</code>	Specifies which test groups to exclude (e.g., <code>-DexcludeGroups=slow</code> ).	Includes all groups

For more options and parameters visit <https://maven.apache.org/surefire/maven-surefire-plugin/index.html>

## Surefire basic parameters - Example

```
mvn clean test -Dtest=MyTestClass
```

```
mvn clean test -Dtest=MyTestClass#myMethod
```

```
mvn clean test -Dgroups=MyTagValue1
```

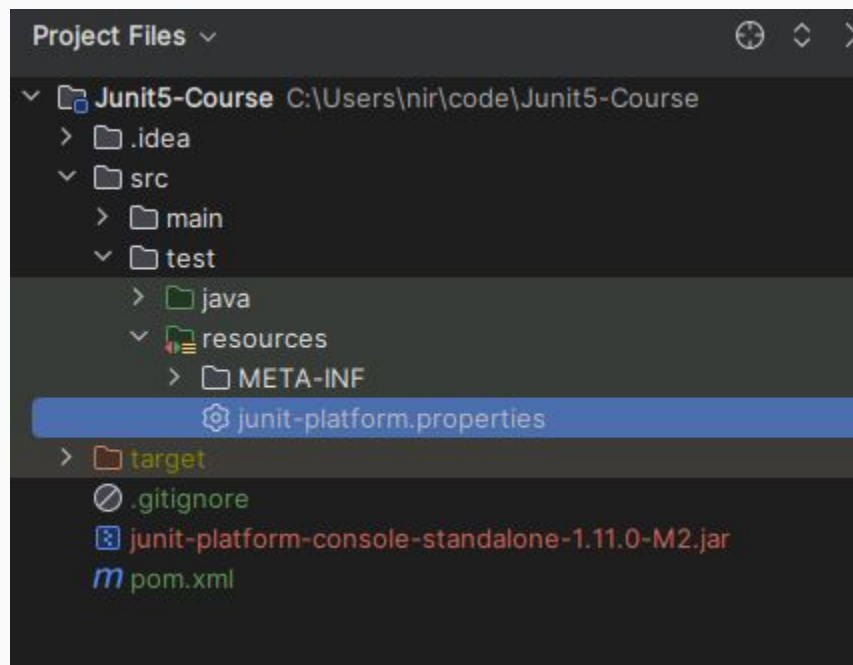
```
mvn clean test -DexcludeGroups=MyTagValue2
```

## Surefire Pass Parameters to Test

```
@Test
public void externalParameterTest() {
    String parameter = System.getProperty("parameter");
    System.out.println("Parameter value: " + parameter);
    assertEquals("MyParameter", parameter);
}
```

```
mvn test -Dtest=AnnotationsTest#externalParameterTest -Dparameter=MyParameter
```

# Parallel Execution



# Parallel Execution Configuration

## Configuration parameters to execute all tests in parallel

```
junit.jupiter.execution.parallel.enabled = true  
junit.jupiter.execution.parallel.mode.default = concurrent
```

.....

## Configuration parameters to execute top-level classes in parallel but methods in same thread

```
junit.jupiter.execution.parallel.enabled = true  
junit.jupiter.execution.parallel.mode.default = same_thread  
junit.jupiter.execution.parallel.mode.classes.default = concurrent
```

.....

## Configuration parameters to execute top-level classes sequentially but their methods in parallel

```
junit.jupiter.execution.parallel.enabled = true  
junit.jupiter.execution.parallel.mode.default = concurrent  
junit.jupiter.execution.parallel.mode.classes.default = same_thread
```

# Parallel Execution Configuration

## Controlling the number of parallel executions

```
junit.jupiter.execution.parallel.enabled = true  
junit.jupiter.execution.parallel.mode.default = concurrent  
junit.jupiter.execution.parallel.config.strategy = fixed  
junit.jupiter.execution.parallel.config.fixed.parallelism = 4
```

# Parallel Execution Configuration

```
@Execution(ExecutionMode.CONCURRENT)
public class ParallelTestExample {

    @Test
    public void test1() throws InterruptedException {
        System.out.println("test #1");
        Thread.sleep(1000);
    }

    @Test
    public void test2() throws InterruptedException {
        System.out.println("test #2");
        Thread.sleep(400);
    }
}
```

[Home](#) » [org.junit.platform](#) » junit-platform-console-standalone



### JUnit Platform Console Standalone

Module "junit-platform-console-standalone" of JUnit 5.

License	<a href="#">EPL 2.0</a>
Tags	<a href="#">junit</a> <a href="#">testing</a> <a href="#">platform</a> <a href="#">console</a>
Ranking	#8765 in MvnRepository ( <a href="#">See Top Artifacts</a> )
Used By	<a href="#">48 artifacts</a>



# JUnit 5 Console Launcher

## Run all tests from this classpath build/classes/java/test

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
  --classpath build/classes/java/test  
  --scan-classpath
```

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
  -cp build/classes/java/test  
  --scan-classpath
```

.....

## Run a specified test with the class name

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
  -cp 'build/classes/java/test'  
  -c com.mk Yong.order.MethodOrderTest
```

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
  -cp 'build/classes/java/test'  
  --select-class com.mk Yong.order.MethodOrderTest
```

# JUnit 5 Console Launcher

## Run all tests from a package

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
-cp 'build/classes/java/test'  
--select-package com.mkyong.order
```

.....

## Run all tests from a package, filter the tests' class name via a regex pattern.

```
$ java -jar junit-platform-console-standalone-1.5.2.jar  
-cp 'build/classes/java/test'  
--select-package com.mkyong.order  
--include-classname='.*Count.*'
```



To be continued...