JUnit (Java Unit Testing) interview questions and answers

BY CHAITANYA SINGH | FILED UNDER: JAVA Q&A

Hello Guys, The below **Junit interview questions** and answers are for both freshers as well as for experienced folks. The reason behind this is that generally interviewers start with the basic questions (fresher level) and go for questions related to advanced topics (experienced level) later. The whole FAQs are divided in two sections. This is the first part.

Must read Q&A second part here: JUnit interview questions

Question: What is unit testing?

Answer: A complete application can be build up by integrating small-2 functional parts, such parts are called as units. It is always better to test such individual units before testing the entire application. The process of testing the functionality and working of these individual unit is known as unit testing. Unit testing can be done manually and the process can also be automated.

Question: Explain Manual testing vs Automated testing?

Answer: Manual testing: Test cases are executed by humans, it's time consuming and costly. **Automated Testing:** No human involvement, test cases are executed by automated tools and programs, It's fast and less costly compared to manual testing.

Question: What is a Unit Test Case?

Answer: Unit test case is nothing but a combination of input data and expected output, which is defined to test the proper functionality of a individual test unit. It's is just to test the behavior of the unit for a particular input data.

Question: Why does JUnit only report the first failure in a single test?

Answer: There is no particular answer for this question, the main reason it does it like this to make the testing process simple and easy. Even though the failures are more than one it only reports the first failure, resolving which may implicitly resolve other unreported failures too.

Question: What is @Test and where it's used?

@Test annotation is used to mark a method as test method, result of which is then compared with expected output to check whether the test is successful or not.

Question: What is @Before and @BeforeClass and it's usage?

@Before annotation:

syntax:
@Before
public void myMethod()

This method should execute before each test. Such methods are generally used for initialization before performing a actual test in test environment.

@BeforeClass annotation:

syntax:
@BeforeClass
public static void myMethod()

This method should execute before all the tests. It executes only once. Method should be declared static. Mostly used for database connectivity tasks before execution of any of the test.

Question: What is @After and @AfterClass and it's usage?

@After annotation:

syntax:
@After
public void myMethod()

This method should execute after each test and used for cleaning up the test and temporary data to avoid memory issues.

@AfterClass annotation:

syntax:
@AfterClass
public static void myMethod()

This method should execute at the end, once all the tests are finished. Method should be declared static and executes only a single time. Mostly used for closing the database connection.

Question: What is @Ignore and when it's used?

@Ignore is used to ignore a test method. It's really useful when we have all the tests in advance but the code is yet to be tested for the particular test, in such scenarios such test methods should be marked with @Ignore annotation.

Question: How will you run JUnit from command window?

Answer:

1) First set the ClassPath as follows:

```
set CLASSPATH=%CLASSPATH%;%JUNIT_HOME%junit.jar
```

2) Invoke JunitCore

```
java org.junit.runner.JUnitCore
```

Question: What is JUnitCore class?

Answer: This class is mainly responsible for executing tests. The org.junit.runner.JUnitCore class has a runClasses() method, which allows us to run one or more test classes. As a output we get a Result (org.junit.runner.Result) object, which we use to filter out the test information.

Question: What is Parameterized test in JUnit and what all annotations used for this?

Answer: Parameterized tests are possible in JUnit and they provides us the liberty of passing parameters into the test classes.

@RunWith(Parameterized.class) – For making a class parametrized.

@Parameters – Parameterized class must have a static method for generating and returning a collection of array, this method should be marked with @Parameters annotation.

Question: What's the use of @Rule annotation?

Answer: @Rule annotation is used for creating objects, which later can be used in test methods.

Question: When you should run JUnit test, Is there any particular time interval between each run?

Answer: No there is no time constraint. A JUnit test needs to run whenever there is a change in the source code. This ensures that the new change passes through all the tests.

Question: Can we change return type of JUnit test method from void to some other type?

Answer: Ideally you should not do this. All the JUnit test methods should have a void return type. If you change the return type then the test method would not be considered as a test method and would be ignored during execution of tests.

Question: Is it possible to pass command-line arguments to a test execution?

Answer: Yes, It’s possible to pass command line arguments to a test execution –

You should use this command:

```
-D JVM command-line options
```

Question: How @Test annotation is used for testing exceptions?

Answer: @Test (expected = Exception.class)

Limitation: It is used for testing only a single exception.