JUnit Interview Questions

Every application that is created by a developer has to be tested for its functionality. JUnit is one such testing platform that uses Java as a mainstream language. If you are a JUnit aspirant, make sure you have studied the questions that could be possibly asked by the interviewer. Here is a list of probable JUnit Interview Questions that can be asked:

Q1. Explain what is JUnit?

JUnit is a testing framework for unit testing. It uses Java as a programming platform, and it is an Open Source Software managed by the JUnit.org community. It uses Java Archive package file format during compilation. The entire package framework resides under different names for different versions. For example, for JUnit 3.8, it is junit.framework package and for JUnit 4 and later, it is org.junit package.

Q2. Mention what is the difference between JUnit and TestNG?

JUnit	TestNG
In JUnit the naming convention for annotation is a bit complicated for, e.g., “Before”, “After” and “Expected”.	In TestNG it is easier to understand annotations like “BeforMethod”, “AfterMethod” and “ExpectedException”.
In JUnit, for a method declaration you have to follow a specific style like using “@BeforeClass” and “@AfterClass”.	In TestNG, there is no restriction like you have to declare methods in a specific format.
In JUnit method name constraint is present.	In TestNG method name constraint is not present, and you can determine any test method names.
JUnit framework does not have “Parameterized Test” or “Dependency Test” features.	TestNG use “dependOnMethods” to implement the dependency testing.
In JUnit grouping of test cases are not available.	In TestNG, grouping of test cases is available.
JUnit does not support parallel execution on Selenium test cases.	In TestNG Parallel execution of Selenium test cases are possible.
It cannot re-run the failed cases.	It can rerun the failed tests.

Q3. Mention different methods of exception handling in JUnit?

Exceptional or analogous conditions during program execution require special processing methods and this exception handling process in a program computation can be of many types. There are different methods of exception handling in JUnit

- Try catch idiom
- With JUnit rule
- With @Test annotation
- With catch exception library
- With customs annotation

Q4. Explain who should use JUnit – a developer or tester? Why you use JUnit to test your code?

JUnit is more often used by developers to implement unit tests in JAVA. It is designed for unit testing that is more a coding process and not a testing process. However, many testers and QA engineers use JUnit for unit testing.
JUnit is used because

- It test early and does automate testing.
- JUnit tests can be compiled with the build so that at unit level, regression testing can be done.
- It allows test code re-usage.
- JUnit tests behave as a document for the unit tests when there is a transfer.

Q5. What are the important JUnit annotations?

The test runner is used to execute the test cases.

- @Test: This is the test method to run first unless otherwise specified.
- @BeforeClass: This is run once before any of the other test methods present in the class.
- @Before: This is run before @Test.
- @After: As the name suggests, this is run after the @Test.
- @AfterClass: This is run one after all of the tests in the class have been run.

Q6. What are the features of JUnit?

There are several features of JUnit such as:

- Open source
- Annotation support for test cases
- Assertion support for checking the expected result
- Test runner support to execute the test case

Q7. What are the useful JUnit extensions?

There are several JUnit extensions such as:

- JWebUnit
- XMLUnit
- Cactus
- MockObject

Q8. Explain what is Unit Test Case?

Unit Test Case is a part of the code that ensures that another part of the code (method) behaves as expected. For each requirement, there must be at least two test cases one negative test and one positive test.

Q9. Explain how you can write a simple JUnit test case?

The simplest way to write a JUnit test case is:

- Determine a subclass of TestCase
- To initialize object(s) under test, override the setup() method
- To release object(s) under test override the teardown() method

Determine one or more public test XYZ() methods that exercise the objects under test and assert expected results.

Q10. Explain what is meant by ignoring test in JUnit?

When your code is not ready, and it would fail if executed then you can use @Ignore annotation.

- It will not execute a test method annotated with @Ignore
- It will not execute any of the test methods of test class if it is annotated with @Ignore

Q11. What are the top advantages of writing unit tests?

The advantages of writing unit tests include Design testability, Code testability and Code maintainability as good unit tests enforces Object Oriented principles such as Single Responsibility etc. which enables people to avoid code smells such as long classes, long methods, large conditionals etc.

Q12. How is code cyclomatic complexity related to unit tests?

As code cyclomatic complexity is determined based on number of decision points within the code and hence execution paths, higher cyclomatic complexity makes it difficult to attain achieve test/code coverage.

Q13. What is mocking and stubbing? Did you use any mocking framework?

Mocking is a feature where an object mimics like a real object. Stubbing are codes that are responsible for taking place of another component.
There are various different Java mocking framework such as Mockito, EasyMock etc.

Q14. What is unit testing method-naming convention that you follow?

The convention you follow should have every information required by the method name in a structured manner. Unit tests name should act like the documentation giving a clear idea of what functionality is tested. There are various techniques that could be used to name unit tests. Some of them are the following: Given… When… Then… Should… etc.

Q15. What do following JUnit test annotations mean?

Use of annotations reduces coding and extra burden from the tester. Following is a list of frequently used JUnit

4 annotations: @Test (@Test identifies a test method)
@Before (Ans: @Before method will execute before every JUnit4 test)@After (Ans: @After method will execute after every JUnit4 test)@BeforeClass (Ans: @BeforeClass method will be executed before the JUnit test for a Class starts)@AfterClass (Ans: @AfterClass method will be executed after the JUnit test for a Class is completed)@Ignore (@Ignore method will not be executed

Thus, these interview questions on JUNIT cover all the major features of JUnit. Some questions are basic while others are on an advanced level. To gain that extra edge over the other candidates, prepare these questions well.