Day36 sept 17

Name :Aravind Kasanagottu.

Id: mvsnarav

Task 3: What is JUnit?

Answer:

JUnit is a popular Java unit testing framework that lets developers write and run repeatable automated tests to verify small pieces of code—typically methods or classes.

• Validates that individual units of code behave as expected.

• Supports test-driven development (TDD) and automated regression testing.

• Provides annotations, assertions, and test runners to simplify testing.

Why use JUnit?

• Automation – write tests once and re-run anytime.

• Regression safety – detect if new code breaks existing functionality.

• Faster development – catch bugs early.

• Integration – works with Maven, Gradle, CI/CD pipelines.

Task 4: Latest Version of JUnit

Answer:

The latest stable JUnit 5 release is \*\*JUnit 5.10.x\*\* (e.g., \*\*5.10.2\*\*).

Task 5: Test Case to Check if Two Strings Are Equal

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

class StringEqualityTest {

@Test

void stringsShouldBeEqual() {

String expected = "hello";

String actual = "hello";

assertEquals(expected, actual, "Strings are not equal!");

}

}

Task 6: Key JUnit 5 Annotations

@Test, @ParameterizedTest, @RepeatedTest, @TestFactory, @TestTemplate, @Disabled

@BeforeEach, @AfterEach, @BeforeAll, @AfterAll

@Tag, @Tags

@DisplayName, @Nested, @Order, @Timeout, @ExtendWith, @TempDir,

@EnabledOnOs, @DisabledOnOs, @EnabledOnJre, @DisabledOnJre, @EnabledIf, @DisabledIf

Task 7: Meta-Annotations vs. Composed Annotations

Meta-annotations are annotations applied to other annotations (e.g., @Retention, @Target).

Example:

@Retention(RetentionPolicy.RUNTIME)

@Target(ElementType.METHOD)

@Test

public @interface MyCustomTest {}

Composed annotations combine multiple annotations:

@Retention(RetentionPolicy.RUNTIME)

@Target(ElementType.METHOD)

@Test

@Tag("smoke")

@DisplayName("Smoke Test")

public @interface SmokeTest {}

Usage:

@SmokeTest

void quickCheck() { /\* ... \*/ }

Task 8: Assertions

Assertions validate that actual results match expectations:

assertEquals, assertNotEquals, assertTrue, assertFalse,

assertNull, assertNotNull, assertThrows, assertAll.

Task 9: Assumptions

Assumptions decide if a test should run. If an assumption fails, the test is skipped.

Examples:

assumeTrue("DEV".equals(env));

assumeFalse(System.getProperty("os.name").contains("Windows"));

assumingThat(env.equals("CI"), () -> assertEquals(2, calc.add(1,1)));

Task 10: Disabling Test Cases

Use @Disabled to temporarily skip a test.

When: unstable tests, unimplemented features, or temporarily ignoring known failures.

Task 11: Conditional Test Execution

@EnabledOnOs / @DisabledOnOs

@EnabledOnJre / @DisabledOnJre

@EnabledIfEnvironmentVariable / @DisabledIfEnvironmentVariable

@EnabledIfSystemProperty / @DisabledIfSystemProperty

@EnabledIf / @DisabledIf

Task 12: Automated Testing

Running tests automatically with frameworks/tools to reduce manual effort and increase reliability.

Task 13: Automated Testing Tools

Unit: JUnit, TestNG, NUnit, PyTest, Mocha/Jest

UI/Functional: Selenium, Cypress, Playwright

API: Postman, RestAssured, Karate

CI/CD: Jenkins, GitHub Actions, GitLab CI/CD

Reporting: Allure, Extent Reports

Task 14: Life Cycle of Test Automation

1. Decide to Automate

2. Tool Selection

3. Test Planning & Design

4. Script Development

5. Execution

6. Result Analysis & Reporting

7. Maintenance

Task 15: JUnit 5 @Suite Example

import org.junit.platform.suite.api.SelectClasses;

import org.junit.platform.suite.api.Suite;

@Suite

@SelectClasses({TestOne.class, TestTwo.class})

public class AllTestsSuite { }

Task 16: Using @Fast and @Slow Tags

import org.junit.jupiter.api.Tag;

import org.junit.jupiter.api.Test;

class TaggedTests {

@Test @Tag("fast")

void fastTest() { /\* quick-running logic \*/ }

@Test @Tag("slow")

void slowTest() throws InterruptedException {

Thread.sleep(2000);

}

}

DynamoDB MCQs

Task 17: How many types of primary keys does DynamoDB use?

\*\*2\*\* (Partition key only, and Partition + Sort key)

Task 18: \_\_\_\_ indexes enable you to query table data using a different key?

\*\*Secondary\*\*

Task 19: DynamoDB uses how many types of secondary indexes?

\*\*2\*\* (Global Secondary Index, Local Secondary Index)

Task 20: Eventually Consistent readings always return current data?

\*\*False\*\*

Task 21: Which action receives a primary key and returns the item’s attributes?

\*\*GetItem\*\*

Task 22: DynamoDB often ensures consistency across all copies in less than a second.

\*\*True\*\*

Task 23: All local secondary indexes retain partition and sort keys from parent tables by default.

\*\*True\*\*