**JUnit Interview Questions**

**1. What is JUnit and why is it used?**

**Answer:**  
JUnit is a unit testing framework for Java. It provides annotations, assertions, and test runners to define and execute tests independently of production code.

**2. What are some important JUnit annotations?**

**Answer:**

* @Test: Marks a method as a test
* @BeforeEach / @Before: Runs before each test
* @AfterEach / @After: Runs after each test
* @BeforeAll / @BeforeClass: Runs once before all tests
* @AfterAll / @AfterClass: Runs once after all tests
* @Disabled: Skips a test

**3. How do you handle expected exceptions in JUnit?**

**Answer:**  
Use assertThrows() or the expected attribute in JUnit 4:

java

CopyEdit

assertThrows(IllegalArgumentException.class, () -> {

myService.method(null);

});

**4. How to test private methods in JUnit?**

**Answer:**

* Refactor to make them package-private
* Use reflection (not recommended for unit tests)
* Prefer testing via public methods that use private logic

**✅ Mockito Interview Questions**

**1. What is Mockito used for?**

**Answer:**  
Mockito is a mocking framework used for writing unit tests in Java by simulating dependencies and verifying interactions.

**2. Difference between @Mock, @InjectMocks, and @Spy?**

**Answer:**

* @Mock: Creates a mock instance
* @InjectMocks: Injects mock dependencies into the object
* @Spy: Partial mock that calls real methods unless stubbed

**3. How to verify a method call in Mockito?**

java

CopyEdit

verify(myDao).save(any(User.class));

**4. How to mock a void method?**

java

CopyEdit

doNothing().when(myService).sendEmail(anyString());

**5. What is the use of ArgumentCaptor?**

**Answer:**  
Captures arguments passed to mocked methods for assertions.

**✅ Spring Test Framework**

**1. What is @SpringBootTest?**

**Answer:**  
It loads the full application context and is used for integration testing in Spring Boot.

**2. Difference between @WebMvcTest, @DataJpaTest, and @SpringBootTest?**

| **Annotation** | **Purpose** |
| --- | --- |
| @WebMvcTest | Test only controller layer |
| @DataJpaTest | Test only JPA repositories |
| @SpringBootTest | Load full context |

**3. How to mock external service in Spring Boot test?**

* Use @MockBean to mock a Spring bean
* Use WireMock or MockRestServiceServer to mock REST calls

**✅ JMeter Interview Questions**

**1. What is JMeter used for?**

**Answer:**  
JMeter is a performance and load testing tool for analyzing and measuring web applications' performance.

**2. What are samplers in JMeter?**

**Answer:**  
Samplers simulate requests like HTTP, FTP, JDBC, SOAP, etc.

**3. What is a thread group?**

**Answer:**  
Defines the number of virtual users, ramp-up time, and loop count in a test.

**4. What is the purpose of assertions in JMeter?**

**Answer:**  
To validate responses (e.g., status code, response contains a string)

**✅ Postman Interview Questions**

**1. What is Postman used for?**

**Answer:**  
Postman is a REST client used to test APIs by sending HTTP requests and analyzing responses.

**2. How do you use environment variables in Postman?**

**Answer:**  
Defined in environments and used as {{variable\_name}}.

**3. What is a collection and how is it useful?**

**Answer:**  
A collection groups related API requests, which can be run manually or via Collection Runner.

**4. Can Postman be used for automated tests?**

**Answer:**  
Yes, using test scripts in the "Tests" tab and Newman CLI for automation.

**✅ MockServer Interview Questions**

**1. What is MockServer?**

**Answer:**  
MockServer is used to mock HTTP and HTTPS requests/responses for testing microservices or client-side interactions.

**2. How does MockServer help in testing?**

**Answer:**  
Allows simulation of responses from a remote server, useful for testing failure scenarios, delays, or 3rd party APIs.

**3. How to write expectations in MockServer?**

java

CopyEdit

mockServerClient

.when(HttpRequest.request().withMethod("GET").withPath("/users"))

.respond(HttpResponse.response().withStatusCode(200).withBody("[...]"));

**4. Difference between WireMock and MockServer?**

| **Feature** | **WireMock** | **MockServer** |
| --- | --- | --- |
| Language | Java only | Java + Node |
| Record/playback | Yes | Limited |
| Use case | API mocking | Service simulation |

**✅ Cross Questions**

**❓ Why not use real services instead of mocks?**

**Answer:**  
Mocking isolates the unit being tested, makes tests faster, more deterministic, and avoids external dependencies.

**❓ Can mocks reduce test coverage quality?**

**Answer:**  
Yes, over-mocking can hide integration issues. Always balance unit and integration tests.

**❓ Why combine JUnit with Mockito?**

**Answer:**  
JUnit provides structure for tests; Mockito helps mock dependencies for isolation and behavior verification.

**✅ Swagger (OpenAPI) Interview Questions & Answers**

**1. What is Swagger?**

**Answer:**  
Swagger (now called OpenAPI Specification) is a framework for describing, producing, consuming, and visualizing RESTful web services.

It includes:

* **Swagger UI** – Interactive API documentation
* **Swagger Editor** – Tool for writing and testing OpenAPI specs
* **Swagger Codegen** – Generates server/client code from spec

**2. What are the main components of a Swagger/OpenAPI document?**

**Answer:**

* openapi: Version of OpenAPI
* info: API title, description, version
* paths: Endpoints and operations
* components: Schemas, security, headers
* servers: Server URLs
* security: Auth requirements
* tags: Grouping for endpoints

**3. How do you integrate Swagger with Spring Boot?**

**Answer:**  
Use **Springdoc OpenAPI**:

xml

CopyEdit

<!-- pom.xml -->

<dependency>

<groupId>org.springdoc</groupId>

<artifactId>springdoc-openapi-ui</artifactId>

<version>1.6.14</version>

</dependency>

Then visit: http://localhost:8080/swagger-ui.html

**4. What annotations are used for Swagger in Spring Boot?**

**Answer:**

* @OpenAPIDefinition
* @Operation – Describes an API method
* @Parameter – Describes input parameter
* @Schema – Describes model
* @ApiResponse – Custom response for status codes
* @Tag – Group endpoints

Example:

java

CopyEdit

@Operation(summary = "Get all employees", description = "Returns list of employees")

@GetMapping("/employees")

public List<Employee> getAll() { ... }

**5. Difference between Swagger 2.0 and OpenAPI 3.0?**

| **Feature** | **Swagger 2.0** | **OpenAPI 3.0** |
| --- | --- | --- |
| Content Types | Only consumes/produces | Supports content object with media types |
| Parameters | Common location | Now split into path, query, cookie, etc. |
| Improved Reusability | Limited | Better via components section |

**6. How does Swagger UI help during development?**

**Answer:**

* Developers and testers can test APIs interactively.
* Serves as live documentation.
* Helps front-end teams understand and test API contracts.

**7. How do you generate client/server code from OpenAPI spec?**

**Answer:**  
Using **Swagger Codegen** or **OpenAPI Generator**:

bash

CopyEdit

openapi-generator-cli generate -i openapi.yaml -g java -o ./generated

**8. How do you handle authentication in Swagger?**

**Answer:**  
Use securitySchemes and security in OpenAPI:

yaml

CopyEdit

components:

securitySchemes:

bearerAuth:

type: http

scheme: bearer

bearerFormat: JWT

security:

- bearerAuth: []

**9. How do you customize Swagger UI (e.g., branding, layout)?**

**Answer:**  
You can:

* Override Swagger UI index.html with custom CSS
* Use Springdoc properties (springdoc.swagger-ui.title, path, etc.)
* Bundle with custom JS if needed

**10. How do you hide endpoints or fields from Swagger documentation?**

**Answer:**

* Use @Hidden on method or controller
* Use @Schema(hidden = true) on model fields
* Exclude controllers via base packages in config

**✅ Cross/Follow-Up Questions**

**❓ Why should teams use Swagger in microservices architecture?**

**Answer:**  
Swagger helps in:

* **Contract-first development**
* **API consistency** across services
* Reducing frontend-backend mismatch
* Enables mocks/stubs generation

**❓ Swagger vs Postman – When to use what?**

| **Feature** | **Swagger** | **Postman** |
| --- | --- | --- |
| API Docs | ✅ (Auto generated) | ❌ (Manual) |
| API Test | ⚠️ (Basic) | ✅ (Advanced) |
| Automation | ✅ with Codegen | ✅ with Newman |
| Use case | Dev + Doc + Contract | Testing + QA |

**❓ Can Swagger be part of CI/CD?**

**Answer:**  
Yes, you can:

* Validate OpenAPI specs in pipeline
* Generate stubs for client/server code
* Publish updated Swagger UI to portal post-deployment