

Ruta de aprendizaje para Java

Usualmente se realiza una ruta de aprendizaje completa:

1. [Computer Science](#) y [Java](#)
2. [Backend Development](#), [Spring + Spring Boot](#) y [SQL](#)
3. [Git & Github](#) y [Systems Administration & DevOps](#)
4. Cloud Computing (ruta depende del proveedor, ejemplo: [AWS](#))

Pero en esta ocasión se redujo la ruta de aprendizaje para conseguir trabajo de la forma más rápida posible como:

Java Testing Developer ó (QA Automation Engineer/SDET Software Development Engineer in Test): techo en salario \$4 millones (sin inglés)

1. ¿Qué es Java y cómo funciona Java?

2. Programación orientada a objetos

3. Sintaxis esencial de Java

- Condicionales: if (bloque/online) & switch (clásico/moderno).
- Iteradores: for, while, do/while, recursividad.
- Declaración: variables, atributos, métodos y constructores.
- Inyección de dependencias: Concepto de desacoplamiento.
- Anotaciones: Metadatos en el código.
- Manejo de errores: try-catch-finally y excepciones.
- Lambdas & Streams: Procesamiento funcional de datos.

4. Motor de testing (JUnit 5)

- Unidad básica: @Test.
- Ciclo de vida: @BeforeAll, @BeforeEach, @AfterEach, @AfterAll.
- Validaciones: Métodos de la clase Assertions
- Clean Code: Uso de nombres descriptivos para los métodos de prueba.

5. Mockito

- Teoría: Diferencia entre dummy, stub y mock.
- Inyección: Uso de @Mock y @InjectMocks.

- Comportamiento: Definición de respuestas (when/thenReturn).
- Verificación: Triggers de verificación (verify).

6. ¿Cómo funciona Spring y Spring Boot? ¿Cómo probarlo?

- Simulación de llamado al API desde afuera (@WebMvcTest).
- Simulación de llamado a base de datos con H2 (@DataJpaTest).
- Inyección de Mocks con @MockBean.

7. Métricas de aseguramiento de calidad

- Uso de JaCoCo para medir cobertura de líneas.
- Identificación y prueba de Edge cases.

8. Git & Github