RESUMEN DIFERENCIAS JUNIT 4 y 5.

Hay muchas diferencias entre JUnit 4 y 5, JUnit 4 es un simple .jar que se puede agregar como librería, sin embargo JUnit 5 es mucho más complejo y tiene 3 módulos, y para pasar de JUnit 4 a Junit 5 es necesario realizar una migración. Por lo que los imports en cada JUnit son los siguientes:

```
import org.junit.jupiter.api.*;      //para JUnit5
import org.junit.*;                 // Para Junit4.
```

Además JUnit 5 permite pruebas dinámicas, es decir, pruebas en ejecución que no se pueden realizar en JUnit 4 que son pruebas estáticas. Otra diferencia importante es que JUnit 5 permite pruebas con parámetros.

Otra característica importante es que JUnit dispone de muchas más anotaciones que JUnit4 y además muchas tienen muchos nombres distintos como se muestra en el código siguiente:

```
public class CalculatorJUnit5Test {

    private Calculator calculator;

    @BeforeEach
    void setUp() {
        calculator = new Calculator();
        System.out.println("Setting up for test");
    }

    @AfterEach
    void tearDown() {
        calculator = null;
        System.out.println("after test);
    }

    @Test
    void testAddition() {
        System.out.println("addition test");
        assertEquals(4, calculator.add(2, 2));
    }
```

Y para Junit 4:

```
public class CalculatorJUnit4Test {

    private Calculator calculator;

    @Before
    public void setUp() {
        calculator = new Calculator();
        System.out.println("Setting up for test");
    }

    @After
    public void tearDown() {
```

```
        calculator = null;
        System.out.println("after test");
    }

    @Test
    public void testAddition() {
        System.out.println("addition test");
        assertEquals(4, calculator.add(2, 2));
    }

    @Test
    public void testSubtraction() {
        System.out.println("subtraction test");
        assertEquals(2, calculator.subtract(4, 2));
    }
}
```

Puede verse que los métodos setUp() es para realizar algo antes de la prueba y el método tearDown es una vez realizada la prueba como indican las anotaciones @BeforeEach y @AfterEach (Estos métodos no son de obligada implementación), y la anotación @Test indica la prueba.

También en JUnit 5 hay muchos más tipos de aserciones o asertos que en JUnit 4, además algunos han cambiado como es el caso de assertEquals que en JUnit 4 el mensaje de error es el primer parámetros y en JUnit 5 es el último parámetro:

```
asertEquals ("mi mensaje", 1, 2)  // JUnit 4.

asertEquals (1, 2, "mi mensaje")   // JUnit 5.
```