

# Apuntes Generados

## Resumen

El contenido proporcionado se enfoca en la utilización de JUnit, un framework de Java para realizar pruebas unitarias en el desarrollo de software. Se detallan las características de JUnit 4, incluyendo la estructura de las pruebas, las anotaciones más usadas, y cómo organizar y ejecutar tests unitarios para asegurar el correcto funcionamiento del código.

## Conceptos Clave

- **JUnit:** Framework de Java para la ejecución controlada de clases Java, permitiendo evaluar si los métodos se comportan como se espera.
- **Test Unitario:** Examinan el comportamiento de una unidad de trabajo en Java, usualmente un método.
- **Test Class/Test Case:** Clase que contiene uno o más métodos anotados con `@Test`, agrupando comportamientos relacionados.
- **Suite/Test Suite:** Grupo de tests relacionados, usualmente incluyendo test classes del mismo paquete.
- **Runner/Test Runner:** Clase que ejecuta las test suites.
- **Asserts:** Métodos de la clase `org.junit.Assert` que permiten verificar relaciones entre los resultados reales y esperados, como `assertEquals`, `assertNotNull`, y `assertTrue`.

## Detalles Importantes

- **Integración en IDEs:** JUnit está integrado en los IDEs más comunes como NetBeans, Eclipse y IntelliJ.
- **Anotaciones:** Las anotaciones como `@Test`, `@Before`, `@After`, `@BeforeClass`, y `@AfterClass` son fundamentales para definir el comportamiento de los tests.
- **Control de Ejecución:** Se pueden controlar aspectos de los tests como el tiempo de ejecución y la generación de excepciones mediante parámetros en la anotación `@Test`.

## Ejemplos y Aplicaciones

1. Clase de ejemplo `Calculator` con métodos para sumar, restar, multiplicar y dividir.

2. Clase de prueba `CalculatorTest` que utiliza asserts para verificar el comportamiento de los métodos de `Calculator`.

3. Uso de `JUnitCore` para ejecutar tests y reportar resultados, mostrando cómo manejar fallos y éxitos en las pruebas.