Pruebas Unitarias

En [programación](https://es.wikipedia.org/wiki/Programaci%C3%B3n), una **prueba unitaria** es una forma de comprobar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Luego, con las [Pruebas de Integración](https://es.wikipedia.org/wiki/Pruebas_de_Integraci%C3%B3n), se podrá asegurar el correcto funcionamiento del sistema o subsistema en cuestión.

La idea es escribir casos de prueba para cada función no trivial o [método](https://es.wikipedia.org/wiki/M%C3%A9todo_(inform%C3%A1tica)) en el módulo, de forma que cada caso sea independiente del resto.

Algunas de las características de las pruebas unitarias son:

* Prueba solamente pequeñas cantidades de código: Solamente prueba el código del requerimiento específico.
* Se aísla de otro código y de otros desarrolladores: El unit test prueba exclusivamente el código relacionado con el requerimiento y no interfiere con el trabajo hecho por otros desarrolladores.
* Solamente se prueban los endpoints públicos: Esto principalmente porque los disparadores de los métodos privados son métodos públicos por lo tanto se abarca el código de los métodos privados dentro de las pruebas.
* Los resultados son automatizados: Cuando ejecutamos las pruebas lo podemos hacer de forma individual o de forma grupal. Estas pruebas las hace el motor de prueba y los resultados de los mismos deben de ser precisos con respecto a cada prueba unitaria desarrollada
* Repetible y predecible: No importa el orden y las veces que se repita la prueba, el resultado siempre debe de ser el mismo.
* Son rápidos de desarrollar: Contrariamente a lo que piensan los desarrolladores –> que el desarrollo de pruebas unitarias quita tiempo – los unit test por lo general deben de ser simples y rápidos de desarrollar. Difícilmente una prueba unitaria deba de tomar más de cinco minutos en su desarrollo.

**Ejemplo:**

**1.**Empezaremos creando la clase que queremos probar. Tendrá el siguiente código:

public class Suma {

    private int num1;

   private int num2;

   public Suma(int n1, int n2) {

       num1 = n1;

       num2 = n2;

   }

   public int sumar() {

       int resultado = num1 + num2;

       return resultado;

   }

}

Como vemos, es una clase muy sencilla que dispone de un método que permite sumar dos números.

**2.** Lo siguiente es crear la clase que nos servirá para probar la clase Suma. Queremos saber si la suma se hace correctamente en tres casos: sumando dos números positivos, sumando dos números negativos y sumando un número positivo y un número negativo. El código será el siguiente: 

public class SumaTest {

   @Test

   public void sumaPositivos() {

      System.out.println("Sumando dos números positivos ...");

       Suma S = new Suma(2, 3);

       assertTrue(S.sumar() == 5);

   }

   @Test

   public void sumaNegativos() {

       System.out.println("Sumando dos números negativos ...");

       Suma S = new Suma(-2, -3);

       assertTrue(S.sumar() == -5);

   }

   @Test

   public void sumaPositivoNegativo() {

       System.out.println("Sumando un número positivo y un número negativo ...");

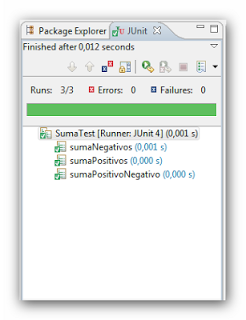
       Suma S = new Suma(2, -3);

       assertTrue(S.sumar() == -1);

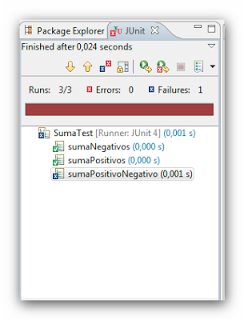
   }

}

**3.** Activamos la vista “JUnit” en Eclipse. Para ello hay que pulsar en: “Window”; “Show View”; “Other...”; “Java”; “JUnit”.   
  
**4.**Y ejecutamos el proyecto como test JUnit, pulsando sobre el proyecto con el botón derecho, luego pulsamos “Run as” y finalmente “JUnit Test”.   
Ahora podremos ver los resultados. Si todo hay ido bien, en la vista JUnit nos aparecerá ésto:

[](http://1.bp.blogspot.com/-V1cZL610nFg/UbA-YqYgwJI/AAAAAAAAAzY/3F64bxIoPnI/s1600/JUnit1.png)

Si ha habido algún fallo nos aparecerá lo siguiente:

[](http://4.bp.blogspot.com/-RUityjkazdk/UbA-cUQVe4I/AAAAAAAAAzg/IL9udB06Wq0/s1600/JUnit2.png)

Como vemos, en el primer caso el test ha sido completado sin fallos. Sin embargo, en el segundo ejemplo (he modificado el código a conciencia para que falle) se ha encontrado un error en “sumaPositivoNegativo”.