**¿Qué es JUnit?**

JUnit se trata de un Framework Open Source para la automatización de las pruebas (tanto unitarias, como de integración) en los proyectos Software. El framework provee al usuario de herramientas, clases y métodos que le facilitan la tarea de realizar pruebas en su sistema y así asegurar su consistencia y funcionalidad.

**¿Cómo se utiliza?**

En primer lugar, es importante que la clase a utilizar tenga la estructura “Test”.

**Tipos de métodos**

* **Método setUp:** Asignamos valores iniciales a variables antes de la ejecución de cada test. Si sólo queremos que se inicialicen al principio una vez, el método se debe llamar "setUpClass"
* **Método tearDown:** Es llamado después de cada test y puede servir para liberar recursos o similar. Igual que antes, si queremos que sólo se llame al final de la ejecución de todos los test, se debe llamar "tearDownClass"
* **Métodos Test:** Contienen las pruebas concretas que vamos a realizar.
* **Métodos auxiliares.**

**Anotaciones**

* **@RunWith:** Se le asigna una clase a la que JUnit invocará en lugar del ejecutor por defecto de JUnit
* **@Before:** Indicamos que el siguiente método se debe ejecutar antes de cada test (precede al método setUp). Si tiene que preceder al método setUpClass, la notación será "@BeforeClass"
* **@After:** Indicamos que el siguiente método se debe ejecutar después de cada test (precede al método tearDown). Si tiene que preceder al método tearDownClass, la notación será "@AfterClass"
* **@Test:** Indicamos a Junit que se trata de un método de Test. En versiones anteriores de JUnit los métodos tenían que tener un nombre con la siguiente estructura: "Test". Con esta notación colocada delante de los métodos podemos elegir el nombre libremente.

**Funciones**

**sertArrayEquals:** Recibe como parámetro dos arrays y comprueba si son iguales. Devuelve assertionError si no se produce el resultado esperado

**sertEquals:** Realiza la comprobación entre dos valores de tipo numérico. Devuelve assertionError si no se produce el resultado esperado

**sertTrue:** Comprueba si una condición se cumple. Devuelve assertionError si no se produce el resultado esperado

**il:** devuelve una alerta informando del fallo en el test.

**¿Qué es una prueba unitaria?**

Es una forma de comprobar la correcta funcionalidad de un fragmento de código. Consiste en aislar una parte de este, y validar el comportamiento como la lógica. Forman parte de una metodología ágil del trabajo y pueden ser realizadas en pocos segundos, mientras que encontrar este error a prueba y error podría llegar a ser una pérdida de recursos.

**¿Por qué son necesarias / obligatorias?**

Nos permiten comprobar que el fragmento de código funciona de forma correcta, mejorar la calidad del proyecto, comprobar la lógica y aumentar la legibilidad del código. Además, nos permiten encontrar errores que al no utilizarlas no podrían ser detectados hasta una fase más avanzada del proyecto, ahorrándonos tiempo y dinero.

**¿Cuál es la librería que se usa en Java para pruebas unitarias?**

Junit: Se trata de un conjunto de bibliotecas que sirven para hacer pruebas unitarias de aplicaciones Java.

**Configurar esa librería en el IDE que usted use habitualmente (Netbeans, eclipse, etc.) Tomar un screeshot donde se vea que esta configurado.**

**Captura de pantalla de un celular

Descripción generada automáticamente**

**Haga un pequeño programa que tenga un método para sumar dos numeros. Luego haga la prueba unitaria para ese método. Tomar un screenshot de esa prueba unitaria.**

Captura de pantalla de computadora

Descripción generada automáticamente

**Bibliografía**

[**https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/248**](https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/248)

[**https://www.discoduroderoer.es/anadir-junit-a-eclipse/**](https://www.discoduroderoer.es/anadir-junit-a-eclipse/)

<https://www.youtube.com/watch?v=XlkMN_i8j5w>

<https://spa.myservername.com/aqui-esta-la-lista-completa-de-juegos-para-sega-genesis-mini-2>