Clase 1 - Qué es Backend

Status	Patrones
Assign	

Es el "detrás de escena", es decir, **el servidor y la base de datos** que ayudan a entregar información del usuario desde una interfaz, hablamos del **back end.**
El **back end** es la parte de la aplicación que **se encarga de toda la lógica para que la misma funcione**.

Algunas de las funciones que se gestionan en esta parte son:

- Las peticiones del **front end.**
- Lógica de negocio.
- Conexión con **bases de datos (relacionales y no relacionales).**
- Logueo de errores, para encontrar luego, más rápidamente las soluciones.
- Uso de **librerías del servidor web**, por ejemplo, para implementar temas de caché o para comprimir las imágenes de la web.
- La **seguridad** de los sitios web que gestiona
- Optimización de los recursos para que las páginas sean performantes.

Un **back end** debe ser capaz de tener una **capa de servicios** para que el **front end** pueda consumirla y así poder realizar peticiones. En el desarrollo de esta capa hay que conectarse a una base de datos y definir que le es permitido mostrar al **front end**.

Tests unitarios vs test integración

`Test unitarios`: cubren una parte del código. De forma PARCIAL.

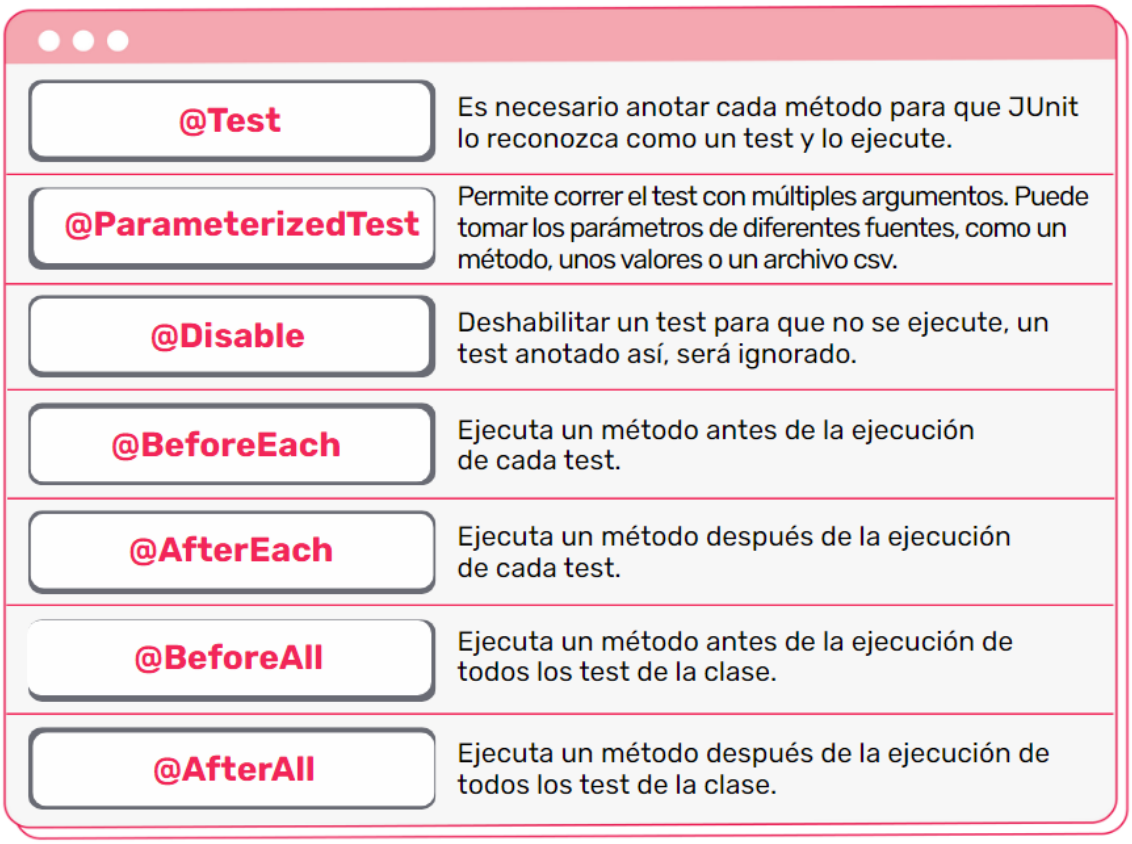
`Test de integración`: manera GLOBAL. Cubren un módulo de tests. Asegurarnos que los diferentes flujos del código funcionan correctamente.

Cuando necesitamos saber si nuestro sistema cumple con las especificaciones o mejorar una parte del código.

Introducción JUnit

¿Para qué se utilizan las siguientes anotaciones?

Te invitamos a arrastrar las mismas con su definición.



💡 **Entre las anotaciones que vimos anteriormente también podemos mencionar a: @Tag: Permite lanzar conjuntos de test en función de las etiquetas que especifiquemos. Anotaciones de ciclo de vida: Sirven para establecer los fixtures. Pueden ser de método o de clase.**

Para concluir

Tal como pudimos dar cuenta, **JUnit** provee una gran variedad de assertions que se encuentran ubicadas en org.junit.jupirter.api.Assertions, por ejemplo:

- assertArrayEquals
- assertEquals : comparar si dos resultados son iguales.
- assertTrue and assertFalse: si el resultado es verdadero o falso.
- assertNull and assertNotNull

- assertSame and assertNotSame
- assertAll
- assertNotEquals
- assertIterableEquals
- assertThrows: asegurarnos que recibimos una excepción.
- assertTimeout and assertTimeoutPreemptively
- assertLinesMatch

```
assertEquals(4, 4);
assertNotEquals(3, 4);
assertTrue(true);
assertFalse(false);
assertNull(null);
assertNotNull("Hello");
assertNotSame(originalObject, otherObject);
```