Clase de Test y Test Double

Test de unidad:

Testeo de la minima unidad de ejecucion/funcionalidad.
En POO, la minima unidad es un metodo.

Objetivo: Aislar cada parte de un programa y mostrar que funciona correctamente.

Cada test confirma que un metodo procude el output esperado ante un input conocido.

Paradoja del testing:

- Escribir casos de testing es deseable.
- Escribir casos de testing es costoso y aburrido.
- Testear TODOS los metodos NO es practico.

Test Double:

- Es necesario realizar pruebas de un SUT que depende de un modulo u objeto.
- El modulo u objeto no se puede utilizar en el ambiente de las pruebas.
- Las pruebas pueden ejercitas:
  - Configuraciones validas del sistema
  - Salidas indirectas del sistema
  - Logica del sistema
  - Protocolos.
  -
  Test Double, es un lenguaje de patrones

El objetivo del test double es:

- Crear un objeto que es una maqueta (polimorfica) del objeto o modulo requerido.
- Utilizar la maueta segun se necesite.
- Se generan diferentes patrones ue se aplican a cada caso.

Patron (Cascaron vacio , Simulacion)

- Test Stub: Cascaron vacio. Sirve para que el SUT envie los mensajes esperados.
- Test Spy: Test Stub + registro de mensajes recibidos.
- Mock Object: Test Stub + verificacion de mensajes recibidos

- Fake object: Imitacion. Se comporta como el modulo real (protocolos, tiempos de respuesta , etc).

Ascensores

- Métodos goUp, goDown, stop
  - Invocan mensajes del “motor”
  - Recibe eventos del “sensor de piso”

Con cada TestDouble se puede:

- **Stub**: recibe los mensajes que envía el Ascensor
  - testear distanceTo(:Request)
- **Spy**: guarda registro de los comando del Ascensor
  - verificar los métodos invocados en el motor
- **Mock**: comprueba la validez de los comandos que envía el Ascensor
- **Fake**: simula el comportamiento del motor → generando eventos del sensor de piso

Test Double

- Implementar clases según sea necesario
  - Es necesario verificar funcionalidad de dependencia
  - Objetos que no están disponibles para probar
- Implementación:
  - Test Stub: simple y barato
  - Fake object: demanda análisis, threading, requiere mantenimiento

Test Doubles:

- Stub : Recibe los mensajes, no hace nada.
- Spy: Guarda registro de mensajes recibidos.
- Mock: Comprueba la validez de los mensajes recibidos.
- Fake: Simula el comportamiento en tiempo y forma.

Explicacion de chatgpt:

Un **Test Double** es un objeto que simula el comportamiento de otro objeto real en un contexto de pruebas. Se usa para aislar una unidad de código (como un método o clase) de sus dependencias y verificar su comportamiento sin depender de esas dependencias reales.

Hay varios tipos de Test Doubles, y cada uno tiene un propósito específico:

1. **Stub:**
   - Objeto que proporciona respuestas predeterminadas a ciertas llamadas.
   - No contiene lógica real, solo devuelve valores fijos.
   - Ejemplo: Un método que siempre devuelve `true` para simular una conexión exitosa a una base de datos.
2. **Spy:**
   - Similar a un `Stub`, pero además registra las interacciones que se hacen con él.
   - Permite verificar si un método fue llamado y con qué parámetros.
3. **Mock:**
   - Un objeto configurado con expectativas sobre cómo debe interactuar el código con él.
   - Verifica si los métodos fueron llamados en un orden específico, cuántas veces y con qué parámetros.
   - Es más estricto que un `Spy`.
4. **Fake:**
   - Un objeto que tiene una implementación funcional pero simplificada del original.
   - Ejemplo: Una base de datos en memoria que simula el comportamiento de una base de datos real.