Mocking

# Testeabilidad

## Conexión

Piensa en un caso dentro de tu código que será difícil (o imposible) de probar utilizando MSTest o JUnit. (Al finalizar la sesión veremos si hemos encontrado la manera de hacerlo)

## Concepto

1. Lectura
2. Responder en parejas las siguientes preguntas

* ¿Qué es la “testeabilidad”?
* ¿Qué relación tiene con el diseño?

## Concreción

* Crear pruebas a un código acoplado
* Responder la hojita
  + Qué problemas han encontrado al crear y ejecutar estas pruebas?
  + Qué atributo de calidad es el más afectado por este diseño?

# Test Doubles

**Inyección e Inversión de Dependencias**

* Concepto: Lectura (Lo puede hacer alguno de los chicos de la clase)
* Concreción: Transformar el código para que use DInv y DInj
* Conclusión:
  + ¿Cuál es el atributo de calidad que se ha visto beneficiado con el principio y patrón anterior?
  + ¿Cuáles son los beneficios de este atributo?

**Test Doubles**

* Concepto: Reemplazar el objeto real por uno más simple.
* Concreción: Transformar el código
* Conclusión: Crea una definición de Test Double
  + Individualmente qué es un Test Double: 1 Min
  + Compartir en Parejas: 1 Min
  + Revisar en internet y comparar la respuesta

# Tipos de Test Doubles y Frameworks

## Concepto

* Definición STUB
* Utilizar una framework en vez del stub manual.
  + De los tests doubles que hicimos, cuál es el stub?
  + Reemplazar por la framework.
* Definición MOCK
* Utilizar una framework en vez del stub manual.
  + De los tests doubles que hicimos, cuál es el stub?
  + Reemplazar por la framework.

## Concreción

* Diferencia entre mock y stub.
  + Parejas: En tus propias palabras explica como diferenciarías un mock de un stub.
* Ejercicio de crear pruebas automatizadas.