CLEAN CODE

CODE SMELLS EN LOS TESTS

Daniel Blanco Calviño

T1: TESTS INSUFICIENTES

- Hacer tests para todas las condiciones y límites de una función. **Se deben probar todas las posibilidades.**
- Mientras no se hayan probado todas las condiciones, los tests son insuficientes.

T2: NO USAR UNA HERRAMIENTA DE COBERTURA

- Las herramientas de cobertura te muestran fácilmente las condiciones y líneas no probadas.
- **SonarQube** te ofrece detalles de **bugs y code smells** presentes en tu código, entre otras muchas métricas.
- Si tu IDE dispone de algún plugin de análisis de cobertura, úsalo! (Ej. **SonarLint**)

T3: EVITAR LOS TEST TRIVIALES

- **No evites escribir test triviales.**
- Son fáciles de implementar y su valor es mayor al coste de producirlos.

T4: TESTS IGNORADOS

- Robert C. Martin: *"A veces, tenemos dudas sobre los detalles de una funcionalidad, porque los requisitos no están claros. Podemos expresar estas dudas con un test comentado, o con un test anotado con @Ignore. La opción que elijas depende de si tu test compila o no"*.

T5: NO TESTEAR LAS CONDICIONES LÍMITE

- Muchos bugs aparecen por no **probar las condiciones límite.**
- Muchas veces probamos nuestro software en condiciones normales, descuidando las condiciones límite.

T6: NO BUSCAR BUGS DE FORMA EXHAUSTIVA

- **Los bugs tienden a estar cerca** unos de otros.
- Si encuentras un bug en una función, revísala, porque es probable que haya más.

T7: LOS PATRONES DE FALLO SON REVELADORES

- Debes **analizar las similitudes entre los fallos de una función.**
- Ejemplo: función que falla cuando le pasamos una cadena de texto con espacios en blanco.

T8: LA COBERTURA DE CÓDIGO ES REVELADORA

- Puedes encontrar el motivo de un fallo en un test analizando las líneas que no se ejecutan.

T9: TESTS LENTOS

- **Si los tests son lentos tendemos a no ejecutarlos.**
- Debemos ejecutar nuestros tests decenas de veces al día, **haz que sean rápidos!**