Todo es una trampa

JESUS ESPINO GARCIA, DEVELOPER

#007000

Kaleidos TAIGA

Introducción

El mundo exterior

Consejo 1:
Adaptadores para bibliotecas externas

Adaptadores

- Nos aislamos de cambios externos.
- Definimos la interfaz que tenga sentido para nosotros
- Adaptamos las bibliotecas para que funcionen con nuestro modelo.

Consejo 2:
Establece contratos para los adaptadores

Contratos de adaptadores

- Debe estar definido como funcionan los adaptadores.
- Estos contratos pueden ser tests, clases abstractas, documentación… (o combinación de estas).
- Preferiblemente tests.
  - Permiten cambiar la biblioteca entera detrás del adaptador y validar que cumple el contrato.

Consejo 3:
Aísla las interacciones con el exterior en repositorios

Repositorios

- Un repositorio es un adaptador que nos aísla de la persistencia.
- El concepto también es aplicable a APIs.
- Nos permite aislarnos de las necesidades de persistencia.
- Pueden ser “plugables”.

Consejo 4:
Establece contratos para los repositorios

Contratos repositorios

- Se aplican las mismas reglas que para los contratos de adaptadores.
- Es especialmente interesante hacerlas con tests para poder probar los adaptadores “plugables”.

Tus dominios

Consejo 5:
Divide y vencerás

Divide y venceras

- Divide tu código en tantos componentes aislados como tenga sentido.
- Piensa que se comunican a través de una API REST (por ejemplo).
- Luego implementa la comunicación según tus necesidades.
- 1 componente de 100.000 líneas es más difícil de mantener y evolucionar que 10 de 10.000.

Consejo 6:
TDD o al menos testea!

TDD

- Idealmente TDD.
- Al menos tests!
- Implementa repositorios para tus tests (rápidos).
- Mockea tus adaptadores (si es necesario).
- Usa un CI para los tests de integración.

Consejo 7:
Escribe Clean Code

Clean Code

- Establece unas reglas mínimas de Clean Code.
- Intenta aplicar todas las que puedas.
- Aquí algunas que yo considero básicas:
  - Reglas de estilo (PEP8)
  - Nombres de variables, funciones y clases con sentido.
  - Funciones pequeñas y con pocos parámetros.
  - Reducir al mínimo los efectos laterales.

Consejo 8:
Cuidado con los frameworks

Frameworks

- Puede o no que sean la mejor solución.
- Suelen ponerse en el centro de nuestra arquitectura.
- Establecen su propia opinión.
- Suele ser difícil llevarlos hasta la frontera de nuestra arquitectura.

¿Por dónde seguir?

Conclusions

- Principios SOLID.
- Clean Code (Robert C. Martin).
- Object Oriented Software Construction (Bertrand Meyer).
- Test Driven Development (Kent Beck)

¿Preguntas?