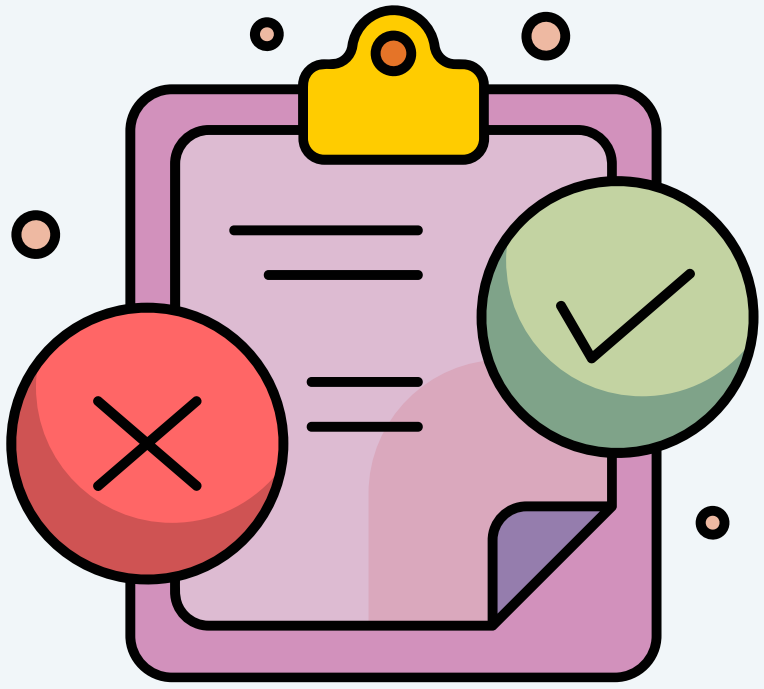
TDD (Test Driven Development)

Desarrollo Guiado por Pruebas: El Camino para Resolver Problemas Reales



PABLO DEL ÁLAMO

Introducción

¿Qué es TDD?

> Test-Driven Development es una técnica de desarrollo que te ayuda a escribir código de calidad y libre de errores.
>
> Enfocado en escribir primero tests antes que el código funcional.

PABLO DEL ÁLAMO

La Filosofía Detrás de TDD

Primero los Tests:

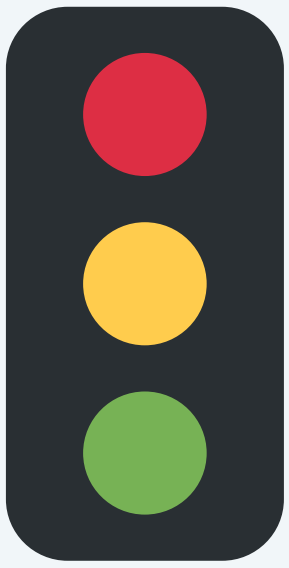
> TDD nos dice que escribamos primero los tests. Así garantizamos que el código cumple con los requisitos desde el principio.

PABLO DEL ÁLAMO

El Ciclo de TDD

Rojo, Verde, Refactor: Este es el mantra.

Escribe un test que falle (Rojo), haz que pase (Verde), y luego mejora el código (Refactor).



PABLO DEL ÁLAMO

Vayamos Paso a Paso

Imagina que estás creando una app de calculadora.

Comenzamos por un test sencillo para una suma.

PABLO DEL ÁLAMO

Escribe un Test que Falla

Código de Ejemplo: assert(sumar(2, 3) === 5);

Este test comprobará si la función sumar devuelve 5 para esos inputs. Debería fallar porque aún no implementamos sumar.

PABLO DEL ÁLAMO

Haz que el Test Pase

Implementa el mínimo código necesario.

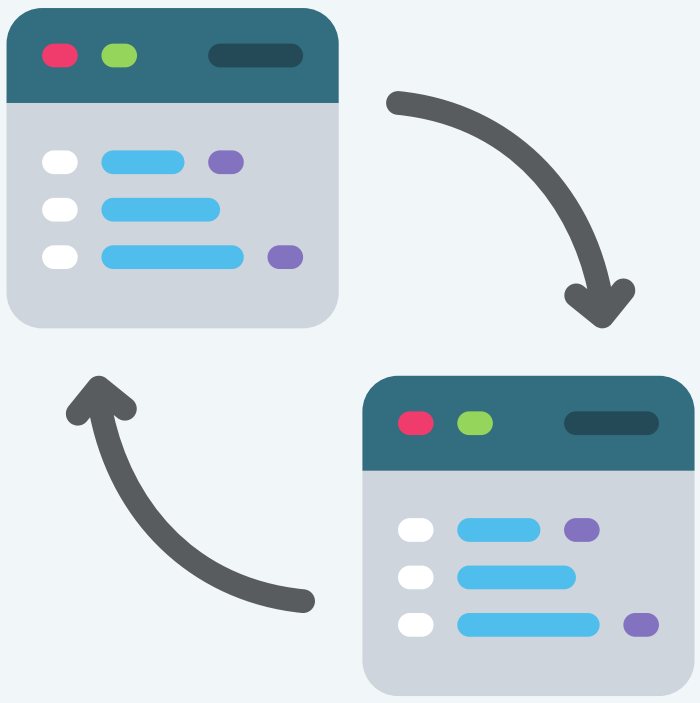
Ejemplo:

```
1 function sumar(a, b) {
2     return a + b;
3 }
```

PABLO DEL ÁLAMO

Refactoriza el Código

Una vez que el test es verde, mejora el código sin modificar su funcionalidad (introduce prácticas de clean code, renombra variables, agrupa partes de código...)



PABLO DEL ÁLAMO

Ventajas de TDD

- Disminución de fallos en el sistema.
- Identificación de necesidades no contempladas.
- Eliminación de fragmentos de código repetidos.
- Reducción en los costes asociados al mantenimiento.
- Optimización del esfuerzo redundante.
- Incremento en la eficiencia del desarrollo.
- Mejora en la generación de documentación.
- Refuerzo de las metodologías ágiles.

PABLO DEL ÁLAMO

Mejora el Diseño del Código

- Pensamiento Orientado al Cliente: TDD fomenta escribir el código pensando en cómo será utilizado realmente, lo cual mejora la experiencia del usuario.

- Modularidad y Flexibilidad: Al escribir tests primero, tiendes a escribir funciones pequeñas que cumplen con tareas específicas, fomentando un diseño modular que es más fácil de mantener y escalar.

PABLO DEL ÁLAMO

Fomenta el Pensamiento Crítico

- Evaluación Constante: Estás continuamente evaluando si tu código pasa las pruebas, lo cual refuerza una mentalidad crítica y analítica.

- Retroalimentación Inmediata: Cada test te da retroalimentación inmediata sobre la funcionalidad, permitiéndote ajustar y mejorar el código de manera iterativa.

PABLO DEL ÁLAMO

Desventajas de TDD

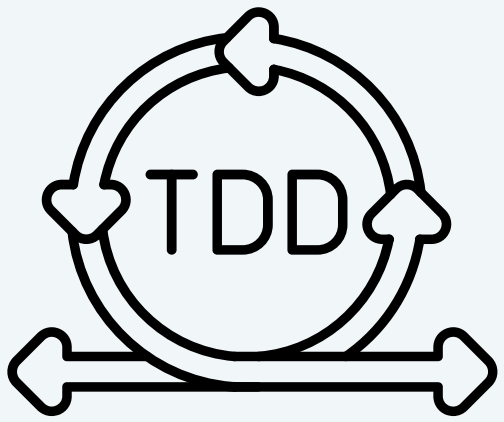
- Curva de Aprendizaje Inicial: Implementar TDD puede parecer pesado al principio; necesitas adquirir destreza al escribir tests eficaces.

- Posibilidad de Tests Frágiles: Mal diseñados, los tests pueden ser frágiles y romperse con cambios menores, requiriendo ajustes frecuentes y potencialmente llevando a más frustración.

PABLO DEL ÁLAMO

¿Cuándo usar TDD?

- Proyectos Nuevos: Es ideal al comenzar nuevos proyectos para establecer una base sólida de calidad desde el principio.
- Desarrollo Crítico: Perfecto para sistemas donde los errores no son una opción, como aplicaciones financieras o de salud.



PABLO DEL ÁLAMO

¿Cuándo NO usar TDD?

- Prototipos Rápidos: Cuando necesitas esbozar un prototipo rápidamente para validar una idea o concepto. La velocidad puede ser más importante que la rigurosidad de los tests.

- Proyectos con Cambios Constantes: En entornos donde los requerimientos cambian de manera drástica y frecuente, mantener los tests puede ser un trabajo enorme y poco beneficioso.

PABLO DEL ÁLAMO

- Aprendizaje de Nuevas Tecnologías: Al principio, cuando estás explorando y aprendiendo un nuevo lenguaje o framework, puede ser más valioso concentrarse en entender los conceptos básicos antes de aplicar TDD.

La clave es saber cuándo TDD aportará valor y cuándo es preferible otros enfoques que se adapten mejor a tus necesidades y contexto de desarrollo.

PABLO DEL ÁLAMO

Herramientas de TDD

- Jest (JavaScript): Popular para aplicaciones frontend con React o Node.js, proporciona una interfaz intuitiva y fácil de configurar.

- Mocha/Chai (JavaScript): Ofrecen gran flexibilidad en cómo estructurar tests y se integran bien con otras bibliotecas.

- JUnit (Java): Es el estándar de pruebas unitarias en el ecosistema Java. Proporciona características robustas para testear y reportar.

PABLO DEL ÁLAMO

TDD y Desarrollo Ágil

- Iteraciones Cómodas: Encaja perfectamente con las metodologías ágiles, mediante iteraciones rápidas y constantes revisiones, asegurando que cada sprint sea eficiente y productivo.

- Valor de Negocio Validado: Los tests aseguran que cada historia de usuario cumple correctamente con la definición de la funcionalidad, alineando el desarrollo con las expectativas del cliente.

PABLO DEL ÁLAMO

Buenas Prácticas

- Mantenlo Simple y Claro: Los tests deben ser fáciles de entender y centrarse en una única función o comportamiento. Divide los casos complejos en partes más pequeñas.

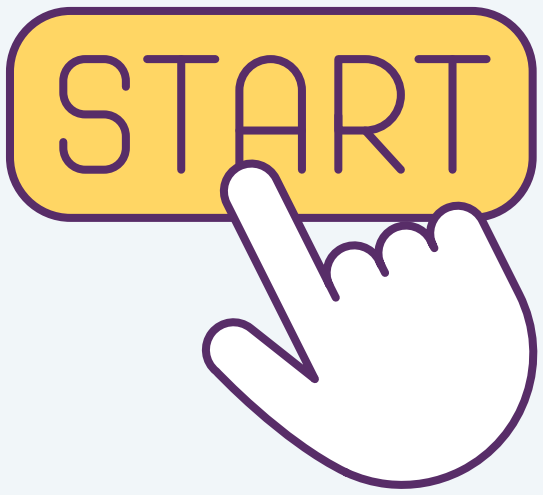
- Revísalo Regularmente: No te quedes con los tests iniciales; refactorízalos junto con el código para mantenerlos precisos y útiles.

PABLO DEL ÁLAMO

Cómo empezar con TDD

- Pequeños Pasos: Empieza añadiendo TDD a nuevas características o módulos pequeños en proyectos actuales, y observa cómo impacta el flujo de trabajo.

- Herramientas Educativas: Utiliza tutoriales, cursos online y comunidades para ayudarte a familiarizarte con las prácticas de TDD.



PABLO DEL ÁLAMO

Mitos de TDD

- Demasiado Rígido: Muchas personas creen erróneamente que TDD limita la creatividad. Sin embargo, proporciona una estructura que permite experimentar sin temor.
- Solo para los Proyectos Grandes: Otro mito es que TDD solo vale para los proyectos enormes y complejos. En realidad, puede ser increíblemente útil en proyectos de cualquier tamaño.

PABLO DEL ÁLAMO

Conclusión

Desarrollar con TDD no solo mejora la calidad del código, sino que transforma la forma en que abordamos el desarrollo.

Al escribir tests antes del código, aseguramos que cada funcionalidad cumpla exactamente lo que se espera, reduciendo bugs y proporcionando estos tests como parte de “documentación”.

Aunque TDD puede parecer desafiante al principio, su adopción ofrece un marco sólido para la innovación, donde los desarrolladores pueden experimentar sin miedo.

Atrévete a implementar TDD en tu próximo proyecto y da paso a la programación segura y eficiente. 🚀

PABLO DEL ÁLAMO

¿Te ha resultado útil?

- Comparte esta guía con tu equipo o amigos desarrolladores.
- Guárdala para tenerla siempre a mano.
- ¡Dale un like o comenta si tienes preguntas!

PABLO DEL ÁLAMO