Clean Code: código limpio, ¿qué es?

hostgator.mx/blog/clean-code-codigo-limpio

Si eres programador, seguramente alguna vez has visto un código mal hecho o con un título que no correspondía de verdad a su función. Este escenario es más común de lo que te puedes imaginar y es justamente el que Clean Code busca combatir. Clean Code, o Código Limpio, es una filosofía de desarrollo de […]

Si eres programador, seguramente alguna vez has visto un código mal hecho o con un título que no correspondía de verdad a su función. Este escenario es más común de lo que te puedes imaginar y es justamente el que Clean Code busca combatir.

Clean Code, o Código Limpio, es una filosofía de desarrollo de software que consiste en aplicar **técnicas simples que facilitan la escritura y lectura de un código**, volviéndolo más fácil de entender.

En este artículo conocerás más sobre esta expresión, incluyendo su origen, sus principios y los beneficios que el Clean Code ofrece en el día a día.

¿Cuándo surgió el término Clean Code?

Las técnicas de código limpio aparecieron por primera vez en el libro "Clean Code: A Handbook of Agile Software Craftsmanship", lanzado en 2008. Este fue escrito por Robert Cecil Martin, conocido en la comunidad como Uncle Bob. El autor trabaja con desarrollo y programación desde 1970 y es uno de los profesionales que lideró el Manifiesto Ágil, en 2001.

Con sus largos años de experiencia, Bob encontró que el problema principal en el desarrollo de software era precisamente la manutención. Es decir, un código mal escrito en su primera versión puede funcionar, pero va a generar grandes problemas en el futuro.

El término se diseminó en la comunidad de forma unánime; y uno de los motivos puede justificarse en el siguiente dato: la proporción media de lectura y escritura de código fuente es de 10 a 1. Esto significa que las personas pasan más tiempo **intentando entender los códigos existentes** de lo que efectivamente creando nuevos.

Con esta información, es posible notar que, además de tener buena arquitectura y buenas prácticas, el código fuente (código principal, que hace que el computador ejecute nuestros comandos) necesita de principios; y esta es la propuesta que Clean Code presenta.

¿Para qué sirve el Clean Code?

Uno de los principales errores que los programadores cometen es creer que, una vez que el código esté listo y funcionando, no necesita de revisiones.

Es importante tener en cuenta que un **sistema nunca está totalmente finalizado**, pues siempre existe la necesidad de realizar actualizaciones y nuevas funcionalidades. Además de esto, el código envejece, y Clean Code se encaja perfectamente en esta situación.

La idea de los principios de Clean Code es convertir el desarrollo web y el mantenimiento del código cada vez más simple.

Un código con muchos ajustes durante mucho tiempo, se vuelve imposible de mantener, por lo que es mejor crear un código nuevo y no continuar con una versión problemática.

Un código limpio evita gastos desnecesarios y prepara al software para las actualizaciones y mejoras.

Conoce las 7 reglas principales del Clean Code

En el libro de Uncle Bob se enumeran buenas prácticas para tener un código limpio. Las principales son:

1 – Los nombres son importantes

La definición de nombre es esencial para el entendimiento de un código. Aquí no importa el tipo de nombre, sean estos:

- Variable;
- Función;
- Parámetro;
- Clase;
- Método.

Al definir un nombre, es necesario tener en mente **dos aspectos principales**:

1. Debe ser preciso y entregar la idea central. Es decir, debe ir directo al punto;
2. No te preocupes por los nombres grandes. Si la función o parámetro necesita de un nombre extenso para demostrar lo que realmente representa, es lo que debes hacer.

2 – Regla del boy scout

Existe un principio que afirma que, si sales del área en que estas acampando, debes dejarla más limpia que cuando la encontraste.

Si traemos esta regla al mundo de la programación, la podemos adaptar como dejar el **código más limpio de lo que estaba antes de editarlo**.

3 – Debes ser el verdadero autor del código

El ser humano está acostumbrado a pensar de manera narrativa, así que el código funciona de la misma forma. Este es una historia y, como los programadores son sus autores, **deben preocuparse por el modo en el que este cuento será presentado**.

Resumiendo, para estructurar un código limpio, es necesario crear funciones simples, claras y pequeñas. Existen dos reglas para crear la narrativa del código:

1. Las funciones deben ser pequeñas;
2. Estas deben ser aún más pequeñas.

No confundas los terminos “nombre” y “función”. Como lo dijimos en el primer principio, los nombres grandes no son un problema, pero las funciones sí.

4 – DRY (Don’t Repeat Yourself)

Este principio puede ser traducido como “no te repitas a ti mismo”. Una expresión, que fue descrita por primera vez en un libro llamado *The Pragmatic Programmer* y se aplica a áreas de desarrollo, como:

- Banco de datos;
- Tests;
- Documentaciones;
- Codificaciones.

DRY defiende que cada parte del conocimiento de un sistema debe tener representación única y **ser totalmente libre de ambigüedades**. En otras palabras, define que no pueden existir dos partes del programa que realicen la misma función.

5 – Comentar solamente lo necesario

Este principio afirma que los comentarios pueden hacerse; sin embargo, estos deben ser necesarios. Según Uncle Bob, los comentarios mienten; y esto tiene una explicación lógica.

Lo que ocurre es que, mientras los **códigos son modificados, los comentarios no**. Estos son olvidados, y por lo tanto, no retratan la funcionalidad real de los códigos.

Entonces, ya sabes; si vas a comentar el código, que sea solamente lo necesario y que sea revisado en conjunto con la versión del código que lo acompaña.

6 – Tratamiento de errores

Hay una frase del autor Michael Feathers, bastante conocida en el área de desarrollo web, que afirma que las cosas pueden salir más; pero cuando esto ocurre, los programadores son los responsables por garantizar que el código continúe realizando lo que necesita.

Es decir, tratar las excepciones de forma correcta es un gran paso para el programador en desarrollo.

7 – Tests limpios

Realizar tests en el área de programación, es una etapa muy importante. Un código solo se considera limpio, después de ser válido a través de pruebas, que también deben ser limpias. Por esta razón, estos deben seguir algunas reglas, como:

- **Fast**: El test debe ser rápido, permitiendo que sea realizado muchísimas veces y en cualquier momento;
- **Independent**: Debe ser independiente, con el fin de evitar que cause efecto cascada cuando ocurra alguna falla, lo que dificulta el análisis de los problemas;

- **Repeatable**: Debe permitir la repetición del test, muchísimas veces y en diferentes ambientes;
- **Self-Validation**: Los tests bien escritos retornan con las respuestas true o false para que el error no sea subjetivo;
- **Timely**: Los tests deben seguir estrictamente el criterio de puntualidad. Además de esto, lo ideal es que sean escritos antes del propio código, pues evita que sea muy complejo para realizar el test.

El Clean Code es un concepto que llegó y se quedó. Sus principios solucionan con eficacia uno de los principales problemas que gran parte de los proyectos de sistemas enfrentan: la manutención.