MODULE 1 - 006: Is Writing Bad Code Immoral for Developers?

This guide discusses a slightly odd question:

`>"Is writing bad code immoral?"*`

It leads us to the concept of the importance of developing well-written code.

But .. the question stills overwhelm: *Is Writing Bad Code Immoral?*

This may seem like a weird question to ask because the mindset of most developers is that code projects are neither **moral** nor **immoral**.
They are simply **programming files** that perform various functionalities.

I would like to think that most developers take pride in their work and therefore want to write code that adheres to **best practices**.
However, given **schedule** and **budget constraints**, many projects devolve, with the top goal becoming "to simply work", and being completed as soon as humanly possible.

But, this mindset can lead to:
- **Missing edge cases** for features - **Poorly organized codebases** that are difficult to maintain

---

A Story Behind this Question

Regarding the question of "Is writing bad code immoral?", I heard a great story from one of my *Computer Science professors* at Texas Tech, Dr. Michael Gelfond.
This is where I got the idea to write this post.

The Story:

A few decades ago, Dr. Gelfond was working as a programmer for a software organization when he ran into a **nasty code bug**.
It took him **several days** to figure out that the previous developer had built a **poorly constructed function**, causing the module he was working on to break.

After telling us this story, Dr. Gelfond posed the question again:
"Is writing bad code immoral?"
Most of the class answered **no**.
Then, he asked:

> **"If we murder someone a few days before they were going to die, is that immoral?"**

The answer was unanimous: **yes**.

He finished his lecture by saying:
> **"Well, wasn't it immoral that the last developer's code stole two days away from my life?"**

My Answer: Absolutely Yes!

That story has stuck with me for years.
Now, my answer to the question **"Is writing bad code immoral?"** is a **resounding yes**!

As developers, we should take **pride** in the work we produce. Not just for our **clients'** or **employers' interests**, but simply because our goal should be to be **true craftsmen** in everything we do.

Coding is the closest thing we have to magic in this world, and I feel **honored** to be able to work with it on a daily basis, along with teaching others how to do the same.

With that in mind, it should motivate us to have a clearly defined goal of being excellent at our craft.

Joe DiMaggio's Wisdom for Developers

One of my all-time favorite baseball players, **Joe DiMaggio**, said:
> *"There is always some kid who may be seeing me for the first time. I owe him my best."*

I try to apply this in all of the code projects I do.
It's easy to fall into **lazy habits**, but I remind myself that **someone might be looking** at this project for the first time, and it could be the **first impression** they have about me as a developer.

If I take shortcuts—even if the application works—it could reflect badly on my work.

Mistakes Are Part of Growth

This doesn't mean that you can't make **mistakes**. Quite the opposite, actually!
I'm constantly striving to become a **better developer**. This means I'm always trying to work on building **features** and **projects** that I haven't created before.

This naturally leads to **mistakes** during the learning process.

However, there is a **clear distinction** between: - Mistakes made while trying to build an **ambitious feature** - **Project bugs** due to laziness and poorly written code.

How to Write Better Code

So, if writing bad code is immoral, what can we do to combat it?
Thankfully, we have a nice set of tools and workflows to help. Here are a few of the ones that I've found to be the most effective:

---

1. TDD/BDD *(Test Driven Development / Behavior Driven Development)*

Regardless of your thoughts on TDD/BDD, there's no denying that when implemented properly, it leads to a **well-constructed codebase**.

- TDD encourages **low coupling** and **small methods**.
- The **refactoring** step ensures that an application is built the **right way**.

I'm a huge fan of TDD to help with building quality code.

---

2. Continuous Integration

With a **comprehensive automated test suite**, continuous integration tools such as **CodeShip** or **Travis** ensure that code won't be pushed to production until it's passed the full test suite.

I've had several occasions where **CodeShip blocked a bad deploy** that would've taken down a site.
Additionally, it provides a **report** to the development team on what needs fixing.

---

3. Pair Programming

This is one of the most powerful tools you can use as a developer.
Pair programming is when you and another developer take turns working on the same project at the same time—preferably in the same room and on the same computer.

Whenever I'm building a **complex feature**, I always use **pair programming** since it's like having **two brains** working on the same function.

---

4. Continuing Education

No matter how long you've been a developer, **learning never stops**.
Each day I try to learn something new—whether it's from **tutorials**, **books**, or **blogs**.

The key is to keep **growing** and always stay curious.

I hope this post has been **thought-provoking** and will help you on your journey toward becoming a **code craftsman**.

Video Lesson Speech

[ENG]

```
# Is Writing Bad Code Immoral for Developers?
This guide discusses a slightly odd question: "is writing bad code immoral?" Which leads to the concept of th
***
This may seem like a weird question to ask because the mindset of most developers is that code projects are n

 I would like to think that most developers take pride in their work and therefore want to write code that ad

However, this mindset can lead to issues such as missing edge cases for features and poorly organized codebas

Regarding the question of "is writing bad code immoral?" I heard a great story from one of my Computer Scienc
 During one of our lectures, he posed the question and then told a story. A few decades ago when he was a pro

It took him several days to figure out that the previous developer had built a poorly constructed function th
After he told us the story he asked us again if writing bad code was immoral, most of the class answered that

That story and question has stuck with me for years and now my answer to the question "is writing bad code im

 As developers, we should take pride in the work that we produce, not just for our clients' or employers' int

One of my all-time favorite baseball players was Joe DiMaggio and he had a great quote that I think is very a

    "There is always some kid who may be seeing me for the first time. I owe him my best."

I try to apply this in all of the code projects that I do, it's easy to fall into lazy habits, however, I rem

This doesn't mean that you can't make mistakes, quite the opposite actually. I'm constantly striving to becom

However, there is a clear distinction between mistakes that get made while you're trying to build an ambitiou

# How to Write Better Code

So if writing bad code is immoral, what can be done to combat it? Thankfully we have a nice set of tools and

    TDD/BDD (Test Driven Development / Behaviour Driven Development)
    Regardless of your thoughts on a test or behavior-driven development, there's no denying that if it's imp

    Continuous Integration
    Assuming that you have a comprehensive automated test suite, continuous integration tools such as CodeShi

    Pair Programming
    This is one of the most powerful tools you can use as a developer. If you're not familiar with it, Pair P
When one of you is coding, the other developer is watching and giving advice or warnings. Whenever I'm buildi
```

Continuing Education
No matter how long you've been a developer, you'll never reach a stage where the learning ends. Each day

I hope that this has been a thought-provoking post and will help you on your journey towards becoming a code

---

[SPA]

¿Es Inmoral Escribir Mal Código?

Este artículo explora una pregunta un tanto inusual: "¿Es inmoral escribir código mal hecho?". Esto nos lleva

Puede parecer una pregunta extraña, ya que la mentalidad de la mayoría de los desarrolladores es que los proy

Sin embargo, dadas las limitaciones de tiempo y presupuesto, muchos proyectos se desvían, y el objetivo princ

Respecto a la pregunta "¿es inmoral escribir código mal hecho?", recuerdo una gran anécdota de uno de mis pro

Durante una de nuestras clases, planteó la pregunta y luego contó una historia. Hace algunas décadas, cuando
Le llevó varios días descubrir que el desarrollador anterior había creado una función mal construida que es
Después de contarnos la historia, nos preguntó de nuevo si escribir código mal hecho era inmoral, la mayoría
Pero luego preguntó si asesinamos a alguien unos días antes de que muriera, "¿es eso inmoral?".
A lo que todos respondieron con un unánime "sí". Terminó su clase diciendo: "Bueno, ¿no fue inmoral que el

Esa historia y pregunta me han acompañado durante años y ahora mi respuesta a la pregunta "¿es inmoral escrib

La codificación es lo más cercano que tenemos a la magia en este mundo y me siento honrado de poder trabajar

Uno de mis jugadores de béisbol favoritos de todos los tiempos fue Joe DiMaggio y tenía una gran cita que cre

"Siempre hay algún niño que puede estar viéndome por primera vez. Le debo lo mejor de mí".

Trato de aplicar esto en todos los proyectos de código que hago, es fácil caer en hábitos perezosos, sin emba

Esto no significa que no puedas cometer errores, todo lo contrario. Constantemente me esfuerzo por convertirm

Sin embargo, existe una clara distinción entre los errores que se cometen mientras intentas construir una fun
Cómo escribir mejor código

Entonces, si escribir código mal hecho es inmoral, ¿qué se puede hacer para combatirlo? Afortunadamente, tene

* **TDD/BDD (Desarrollo Guiado por Pruebas / Desarrollo Guiado por Comportamiento)**:
Independientemente de sus opiniones sobre el desarrollo guiado por pruebas o comportamientos, no se puede neg

* **Integración Continua**:
Suponiendo que tiene una suite de pruebas automatizada completa, las herramientas de integración continua co

* **Programación en Pares**:
Esta es una de las herramientas más poderosas que puede usar como desarrollador. Si no está familiarizado con
Cuando uno de ustedes está codificando, el otro desarrollador está observando y dando consejos o advertencias
Siempre que estoy construyendo una función compleja, siempre uso la programación en pareja, ya que es similar

* **Formación Continua**:
No importa cuánto tiempo haya sido desarrollador, nunca llegará a una etapa en la que termine el aprendizaje.

Espero que este haya sido un artículo que te haya hecho reflexionar y te ayude en tu camino hacia convertirte