10 Waste-of-Time Practices Programmers Should Avoid

#Programming

Introduction:

Boom Shakalaka!! Let's Begin Horses :)

We all thought of programming perceive as a miracle sword on bringing the finest gypsy-sorcerers to the world! Unfortunately, it doesn't seems that way and therefore, we are just human with lack of logic, creativity and persistence but programming ain't one.

Nonetheless, programmers often find themselves as a maze puzzle-solver that requires deep-critical thinking of problem-solving and more experiences and practices to open the hidden treasure.

Here are Top 10 Waste-of-Time Practices Programmers Should Avoid:

1. **Code Comments** -- **The Misunderstood Monster**

-- *Code comments should serve you as a first-aid kit, not a written novel that brings confuses to the scholar of programmer.*
-- *Complex comments lead to distraction and confusion especially you write an comment on every line of code which gives poor structured of code that's harder to understand.*

Important Notes

- Comments should gives right context and value to the explanation of '**why**' the code works and '**how**' does it operate in its own way.

2. **Perfectionism** -- **The Illusionist**

-- *Code aren't made to be perfect! It's made to be **readable**, **maintainable**, and **efficient**.*
-- *Premature optimization is another example of the perfectionism family.* Exp: Trying drive R34 GTR to go high-speed Genting but you don't even know how to control the speed itself.
-- *Instead, focus on getting the right functionality first, then optimize if is necessary to upgrade.*

Important Notes

- **Readability and efficiency** will always win the game!

3. **Ignoring the Newbies (Non-Programmers)**

-- *Sometimes, being collaborative teamwork effort with newbies (**project managers, UX/UI designers, clients**) can remove unnecessary time of digging the right problems.*
-- *Always treat non-programmers as your **fellow partner/side-kick** and teach him/her as part of your mentor.*

Important Notes

- **Giving more clarity** of your objectives to your fellow side-kick could smoother your programming journey as developer.

4. **Not Leveraging Magical Tools (Existing Libraries and Frameworks)**

-- *Always leverage the existing libraries and frameworks can save precious time and offer more robust and efficient solution to the problem.*
-- *But be careful, it's essential to understand the magic behind these magic tools. Lack of knowledge and understanding can hijack the system and breaks the system.*

Important Notes

- Always understand theses tools to prevent any **troubleshooting issues** and **increase more potential** to your coding ability.

5. **Neglecting Self-care**

-- *Programmers ain't **robots/NPC** dude!*
-- *Programmers are human too. We need tons of break and enjoy the freedom that we want to enjoy besides work.*
-- *Programming isn't just about writing codes and compile it. It involves **creative problem-solving**, which require **well-rested** and **healthy mindset**.*
-- ***Burnout** is one of the most common issues. It is a sign that you need to step back, relax and recharge your Power Supply.*

Important Notes

- Always be kind to yourself, take breaks, exercise, socialize and don't forget to have a life and enjoy the happiness of freedom.
- A billionaire once say, one of the investment you should look out for is your **health!** Not worthy if you are dead mole.

6. **Skipping the Planning Phase**

-- *Designing a good strategy mindgame may seem like a long-process, but it save you in a long run without frustration.*
What differs from master sorcerer with a novice wizard?
-- *Sorcerer meticulously prepares his portion ingredients before beginning his ritual experiment. Same goes to programmer.*

Important Notes

- It not about forecasting every possible scenario but to create more flexible framework that can go along with potential curveballs.
- **organize your code structure**, **define your functions and classes**, and have **clear sense of direction** that can helps you to navigate the better road through development process.

7. **Debugging Without Understanding the Problem**

-- *Always ensure you fully grasp the understanding of the issue by reproduce the bug, gather information and try understand the problematic may occurs unexpectedly.*

-- *Imagine you in a duel match, you open your ability without understanding your opponent's counterattack ability. Sounds booming right! Similar to debugging as well.*

Important Notes

- Embrace detective's mindset when debugging.
- Always gather the clues behind this mystery criminal case, leading you closer to the evidence and expose the truth behind it.

8. **Overlooking Code Reviews**

-- *Code reviews not only improve code quality but also spread knowledge within the team, promoting learning growth outcome.*

-- *However, fail to notice towards code reviews is like ignoring your parent's advise on preventing the bad habit that could harm your life.*

-- *Therefore, your team members' experience can help detect overlooked bugs, optimize code, and discover better ways on tackling the problem.*

Important Notes

- Code reviews are a treasure lover for learning curiosity and enhancement opportunity in team bonding.
- Famous quote once said: "It takes a great deal of bravery to stand up to our enemies , but just as much to stand up to our friends."
- Always request code reviews to learn from the feedback and learn from it.

9. **Not Taking Advantage of Automation**

-- *Imagine you have tons of robots at your house handling explicitly tasks while you focus on the tasks that brings your expertise needs.*

-- *In the world of programming, utilizing automation tools not only saves time but also reduces risk of human error.*

Important Notes

- Automated testing tools (JUnit / Selenium)
- CI/CD tools (Jenkins/ Travis CI)
- Code formatting tools (Prettier / ESLint)
- These tools can **enhance your productivity** and they are your **loyal housekeeper.**

10. **Multitasking and Context Switching**

-- *Focusing on one task at a time, and avoid constant context-switching between different parts of your project.*

-- *Although the process is slower, serotonin chemical will boost the productivity once the task is done.*

-- *Multitasking might make you feel super productive sorcerer, but research shows that it reduce productivity and leads more errors time to time.*

Important Notes

- It takes time to "load" all the relevant information tasks into your brain.
- Switching too often will result wasting potential time by just loading and unloading the guns rather than progressing the target.
- Always channel your inner brain, and focus on your own magical forces (full-force attention) on one particular task in a single period.

Conclusion

- **Treat code as a form of art the way science treats nature as a way of life.**
