DWA_07.4 Knowledge Check_DWA7

---

1. Which were the three best abstractions, and why?

Interface Segregation Principle - it makes the web page more user friendly without confusing the user with unnecessary information and requirements.

Single-Responsibility Principle - it makes the code base more readable to other collaborators and developers looking to understand your code and also helps you modify the code without ruining other parts of the code.

Dependency Inversion Principle - it helps with making sure data is not tampered with by low level functions making security a bit better by not accessing information directly or modifying it.

---

2. Which were the three worst abstractions, and why?

Liskov Substitution Principle - it often would confuse other developers who are working on a deadline by mistakenly calling/modifying or even removing the wrong class be it a superclass or subclass

Open-Closed Principle - it makes it difficult to simply make bug fixes that just need alterations to the source code instead of adding more code making it more complex

---

3. How can The three worst abstractions be improved via SOLID principles.

Liskov Substitution Principle - by naming classes by groups or names that would not confuse other developers working on the code

Open-Closed Principle - by documenting what should or is open to modification or deletion

---