DWA_07.4 Knowledge Check_DWA7

---

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

1. **Code Abstraction**
   By abstracting code into modules, classes, or functions, developers create reusable, understandable building block
2. **Conceptual Abstractions**

   These higher-level abstractions help players strategize without getting bogged down in specific move details.

3. **Language-Level Abstraction**

   These language constructs simplify development by encapsulating common patterns and behaviors.

---

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

Over-Engineering and Premature Abstraction:

- **Issue**: Sometimes, developers overcomplicate their code by abstracting too early or creating unnecessary layers of complexity.

Lack of Native Abstraction Support in JavaScript:

- **Issue**: JavaScript lacks native support for abstract classes and interfaces.

Choosing the Wrong Abstraction Level:

- **Issue**: Selecting an inappropriate level of abstraction can lead to inefficiencies or code that’s hard to maintain.

---

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

Single-Responsibility Principle (SRP)
Ensure that each abstraction (class, module, etc.) has only one reason to change

Open-Closed Principle (OCP
Design abstractions to be open for extension (new features) but closed for modification (existing code remains untouched)

Liskov Substitution Principle (LSP)
Solution: Derived classes should be substitutable for their base classes without altering program correctness

---