

A Guide to Common Design Patterns in JavaScript

Introduction

Design patterns are reusable solutions to common programming problems. Here are some essential patterns all JavaScript developers should know:

What are Design Patterns?

- Proven development paradigms for solving issues.
- Make code more maintainable and scalable.
- Examples include MVP, MVC, and more architectural patterns.

The Factory Pattern

- Define an interface for creating objects.
- Helps encapsulate complex creation logic. Example: shape factory for generating shapes.

The Observer Pattern

- One-to-many relationship between objects.
- Observers are notified of changes by subjects. Example: event listeners updating on state change

The Singleton Pattern

- Restricts class instantiation to one object instance.
- Useful for situations requiring one centralized manager. Example: limiting app to single database connection

Declarative vs Imperative

- Declarative focuses on desired outcome.
- Imperative focuses on specific steps to achieve it.
- Finding the right balance boosts productivity.

Avoiding Anti-Patterns

- Common bad practices like god objects or spaghetti code.
- Can lead to unmaintainable and unreliable systems.

Conclusion

Applying standard design patterns helps build stable JavaScript applications.