

SOLID React

Ethan Kent

Spoonflower

August 21, 2023

SOLID Principles

Single-Responsibility Principle Functions (etc.) should have exactly one reason to change/be responsible to exactly one actor or stakeholder.

Open–Closed Principle Functions (etc.) should be open for extension but closed for modification.

Liskov Substitution Principle You should be able to replace a type with its subtype.

Interface-Segregation Principle Code should not be required to rely on methods or properties it doesn't use.

Dependency-Inversion Principle Functions (etc.) should depend on abstractions rather than concretions.

Applying SOLID to React

- Single-Responsibility Principle
 - Separate logic from rendering.
 - Consider volatility when designing shared code.
- Open-Closed Principle
 - Share code by composing rudimentary components; don't use conditional logic to specialize a kitchen-sink component.
 - Invert dependencies.
- Liskov Substitution Principle
 - Use TypeScript or strict type checking to ensure compatibility.
- Interface-Segregation Principle
 - Avoid prop-drilling (also relevant to OCP).
 - Avoid using optionality to facilitate polymorphism (also relevant to OCP).
- Dependency-Inversion Principle
 - Use services/Onion Architecture to define props to allow polymorphism (also relevant to OCP).