

SOLID Design Principles

How do I make my software last through changing requirements...usage patterns...and technology?



S

Single responsibility principle

A class (or method) should have only a single responsibility; have one and only one reason to change

O

Open/closed principle

Objects or entities should be open for extension but closed for modification.

L

Liskov substitution principle

Objects should be replaceable with instances of subtypes without altering program correctness

I

Interface segregation principle

A client should never be forced to implement an interface that it doesn't use, or clients shouldn't be forced to depend on methods they do not use.

D

Dependency inversion principle

Entities must depend on abstractions, not on concretions. A high-level module must not depend directly on a low-level module.

The one you'll hear the most in this workshop...

How do I make my software last through changing requirements...usage patterns...and technology?

S Single responsibility principle

A class (or method) should have only a single responsibility; have one and only one reason to change

