SOLID software design principles

|  |  |  |
| --- | --- | --- |
|  | PRINCIPLE | DESCRIPTION |
| S | Single responsibility | A class should have one, and only one, reason to change, which means it should have only one function. |
| O | Open/closed | Software objects should be open to extension, but closed for modification. |
| L | Liskov substitution | Objects of the same type should be replaceable with others from the same category without altering the function of the program. |
| I | Interface segregation | No client should be forced to depend on methods it does not use. The program’s interfaces should always be kept smaller and separate from one another. |
| D | Dependency inversion | High-level modules should not depend on low-level modules, but both should depend on abstractions. While abstractions should not depend on details, details should depend on abstractions. |