**SOLID**

SOLID se compone de una serie de principios y de buenas prácticas que se deberían tener como base antes de proponer una arquitectura de software para el desarrollo de nuestras aplicaciones.

Los principios SOLID nos permiten administrar la mayoría de los problemas de diseño de software, y si seguimos los mismos, vamos a conseguir desarrollar un código más limpio, más mantenible, más escalable a futuro y menos propenso a errores.

SOLID es un acrónimo, y cada una de las letras que lo componen tiene un significado:

* S: se refiere al Principio de Responsabilidad Única (Single Responsibility Principle), que en resumidas cuentas lo que dice es que cada módulo de software debe tener una única razón para cambiar.
* O: hace referencia al Principio de Abierto/Cerrado (Open/Closed Principle), que dice que el código debería estar abierto para extenderlo y para añadirle nuevas funcionalidades, pero en cambio debería estar cerrado a modificaciones, salvo las modificaciones que se deban realizar si se encuentra algún error.
* L: viene a referirse al Principio de Sustitución de Liskov (Liskov Substitution Principle), que en resumen lo que dice es que una clase derivada debe poder ser sustituida por su clase base.
* I: se refiere al Principio de Segregación de Interfaces (Interface Segregation Principle), que dice que se han de utilizar interfaces con propósito específicos, o sea que tengan responsabilidades únicas y que se piensen bien y no se hagan interfaces grandes.
* D: hace referencia al Principio de Inversión de la Tendencia (Dependency Inversion Principle), que para conseguirlo se hace uso de la inyección de dependencias.