SOLID Introduction

1. SOLID principles are kind of a design principals.They help to manage most of the software design principles.
2. There are five design principles that used to develop software more understandable, flexible and maintainable.
3. SOLID principles are designed by Robert C. Martin
4. Michael Feathers introduced the SOLID acronym.

Five SOLID Principal

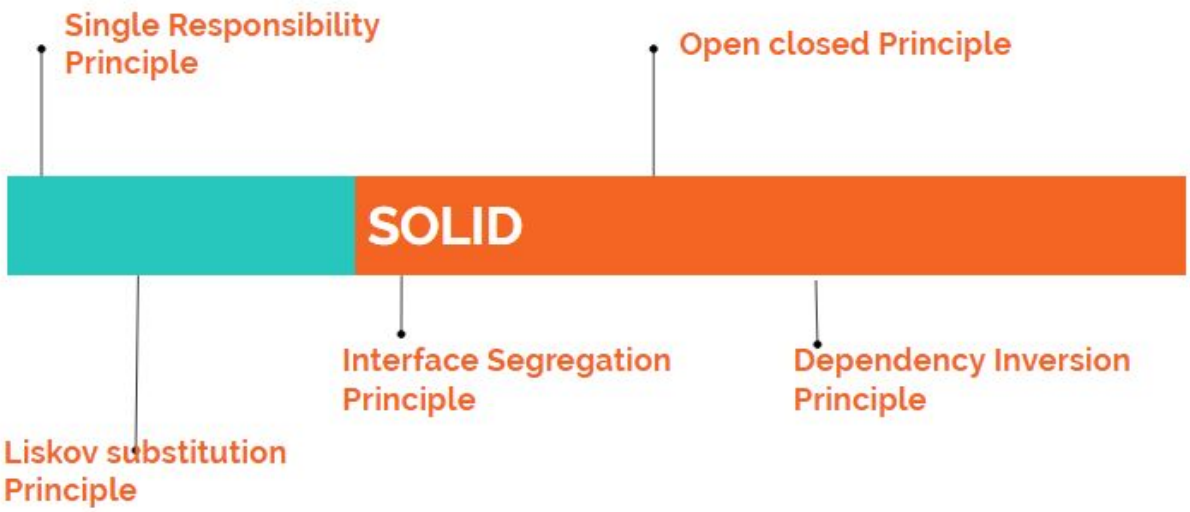

Figure 1.0

Explanation of SOLID principles

Single Responsibility Principle

One Class and one member must do the one task.That task should not do the multiple tasks at a given time.**One responsibility** must be done by the given class or member.

Liskov Substitution Principle

In programme language we use common functionality in the project,To gain this common functionalities we use the inheritance,Then there must be a relationship between child and parent.This concept shows the way to do it.It tells What type of technoliest should we use and what standards that we should use.

Open/Closed Principle

When we are adding new functionality.we add new classes,methods and attributes.We do not add that functionality to the same class or method.Some time requirement may changes.Therefore we don't change the existing functionality .We have to implement new logic for this.

If we implement a new logic to the same class.Maintenance will be difficult,There may be a multiple duplicate code.Testing may be difficult.

Interface Segregation Principle

This concept talks about the interfaces in OOP concept.We can use multiple interfaces without using the big ,complex interface.We can break the interface to the small interface throughout the project.We can use them as sub modules.

Dependency Inversion Principle

This concept talks about the high level module and the law level module.High level module does not depend on the low level modules.There is no direct connection between low level module and the high level module.There is an agent between them.That agent keep the communication between them.As example There is a middle layer between business layer and data access layer .Therefore we can achieve loose coupling feature in programming language.

Advantages of SOLID Principles

1. Can achieve loose coupling
2. Can achieve testing
3. Reduce the complexity of code.
4. Increase the testability

If we don’t follow SOLID Principles we

1. Maintenances may be difficult
2. There are duplication code.
3. Create new bug when fixing bugs
4. There are new unknown issues.
5. Testing is harder.

References

http://csharp-video-tutorials.blogspot.com/2017/11/solid-design-principles-introduction.html