**DRY:**

Do not repeat yourself!

**What is SOLID Design Principles**

SOLID design principles are arguable the most popular design principles for object-oriented software development.

* **S**ingle Responsibility Principle (SRP)
* **O**pen/Closed Principle (OCP)
* **L**iskov Substitution Principle (LSP)
* **I**nterface Segregation Principle (ISP)
* **D**ependency Inversion Principle (DIP)

**1. Single Responsibility Principle (SRP)**

A class should have only one responsibility hence only a single purpose.

**Benefits of SRP:**

* Easier to maintain
* Easier to understand
* Changed less frequently
* Easily and thoroughly testable

**2. Open / Closed Principle (OCP)**

A class should be open for extension but closed for modification.

**Benefits of OCP:**

* Not allowing modification provides the advantage of not introducing bugs
* All dependent classes will not need to adapt

Note: In an essence if you are using interface, you are using Open/Closed principle.

**3. Liskov Substitution Principle (LSP)**

A subclass should be substitutable by is base class without having any negative impact to the caller.

**Benefits of LSP:**

* Callers does not get surprising behavior when substitution applies.
* Complex bug which might arise due to conflicting behavior between inheritance is avoided easily.

**4. Interface Segregation Principle (ISP)**

The interface segregation principle is all about separating interfaces. Basically multiple specific interfaces are better than generic single interface.

**Benefits of ISP:**

* Same as SRP, easier to manage and maintain.
* Works hand in hand with SRP.

**5. Dependency Inversion Principle (DIP)**

According to the Dependency Inversion Principle, classes should only depends on contracts, meaning interfaces or abstract classes rather than concrete implementation.

As you can see, this is very closely related to Open/Close Principle.