1. SOLID

2. MULTITINAT

3. AUTH 2

4. SPRING Security

5. Design pattern

6. System Design

SOLID

They all serve the same purpose:

"To create understandable, readable, and testable code that many developers can collaboratively work on."

Let's look at each principle one by one. Following the SOLID acronym, they are:

* The **S**ingle Responsibility Principle
* The **O**pen-Closed Principle
* The **L**iskov Substitution Principle
* The **I**nterface Segregation Principle
* The **D**ependency Inversion Principle

## The Single Responsibility Principle

The Single Responsibility Principle states that **a class should do one thing and therefore it should have only a single reason to change**.

## Open-Closed Principle

The Open-Closed Principle requires that **classes should be open for extension and closed to modification.**

## Liskov Substitution Principle

The Liskov Substitution Principle states that subclasses should be substitutable for their base classes.

This means that, given that class B is a subclass of class A, we should be able to pass an object of class B to any method that expects an object of class A and the method should not give any weird output in that case

## Interface Segregation Principle

Segregation means keeping things separated, and the Interface Segregation Principle is about separating the interfaces.

The principle states that many client-specific interfaces are better than one general-purpose interface. Clients should not be forced to implement a function they do no need.

## Dependency Inversion Principle

The Dependency Inversion principle states that our classes should depend upon interfaces or abstract classes instead of concrete classes and functions.

In his [article](https://fi.ort.edu.uy/innovaportal/file/2032/1/design_principles.pdf)(2000), Uncle Bob summarizes this principle as follows:

"If the OCP states the goal of OO architecture, the DIP states the primary mechanism".

These two principles are indeed related and we have applied this pattern before while we were discussing the Open-Closed Principle.

Topics: O-Auth

Topics will be covered:

1. Definition of Authentication and Authorization.
2. Definition of O-AUTH
3. Some Definition(Principles)
4. Work Flow
5. Example
6. Advantage
7. Disadvantage

Authentication: **Authentication verifies the identity of a user.**

Authorization: **Authorization determines their access rights.**

[OAuth 2.0](https://tools.ietf.org/html/rfc6749), which stands for “Open Authorization.

Open Authorization is Authorization Framework that allow to an application to access resources hosted by other web apps on behalf of a user.

একটি অ্যাপ্লিকেশন কে কিছু resources এর access দেয় অন্য একটা hosted application থেকে একটা user এর পক্ষ থেকে ।

Invasion and Version:

O-auth 1.0 started by twitter in 2006

O-auth 2.0 In January 2013, the Internet Engineering Task Force Introduce 0-auth 2.0.

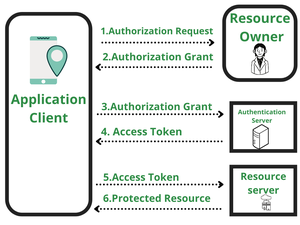
Some Definition (Principles):

## Principles of OAuth2.0: OAuth 2.0 is an Authorization protocol and it is not a authentication protocol.

Principles:

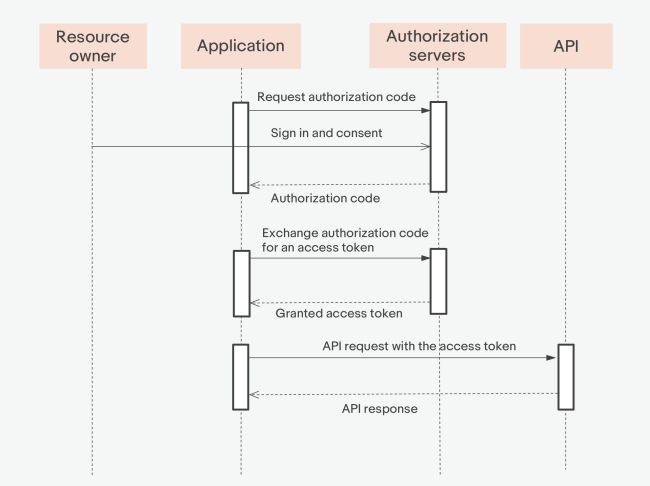
* **Resource Owner**: The user or system that owns the protected resources and can grant access to them.
* **Client**: The client is the system that requires access to the protected resources.
* **Authorization Server**: This server receives requests from the Client for Access Tokens and issues them upon successful authentication and consent by the Resource Owner.
* **Resource Server**: A server that protects the user’s resources and receives access requests from the Client.

Work Flow:



1. The client requests authorization by directing the resource owner to the authorization server.
2. The authorization server authenticates the resource owner and informs the user about the client and the data requested by the client. Clients cannot access user credentials since authentication is performed by the authentication server.
3. Once the user grants permission to access the protected data, the authorization server redirects the user to the client with the temporary authorization code.
4. The client requests an access token in exchange for the authorization code.
5. The authorization server authenticates the client, verifies the code, and will issue an access token to the client.
6. Now the client can access protected resources by presenting the access token to the resource server.
7. If the access token is valid, the resource server returns the requested resources to the client.

Authorization Grand flow:



Advantage:

* Capability to authorize a user.
* OAuth 2.0 is a very flexible protocol that relies on SSL (Secure Sockets Layer) to save user access token.
* This protocols ensure data safety.
* It has ability to share data for users without having to release personal information.
* It is easier to implement and provides stronger authentication.

Disadvantage:

* There is no common format, as a result, each service requires its own implementation
* When a token is stolen, an attacker gains access to the secure data for a while.