SOLID: Interface Segregation Principle

Get introduced to the Interface Segregation Principle.

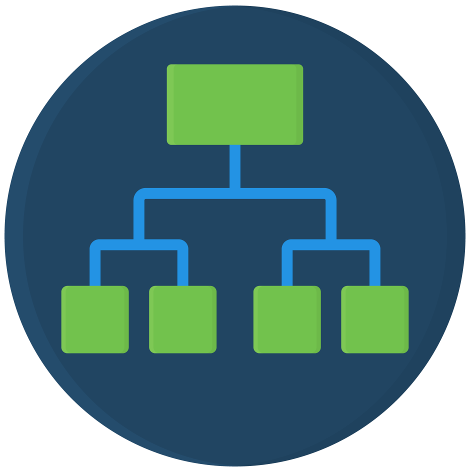
We'll cover the following

- Introduction
- Example
  - Violation
  - Solution
- Conclusion

Introduction

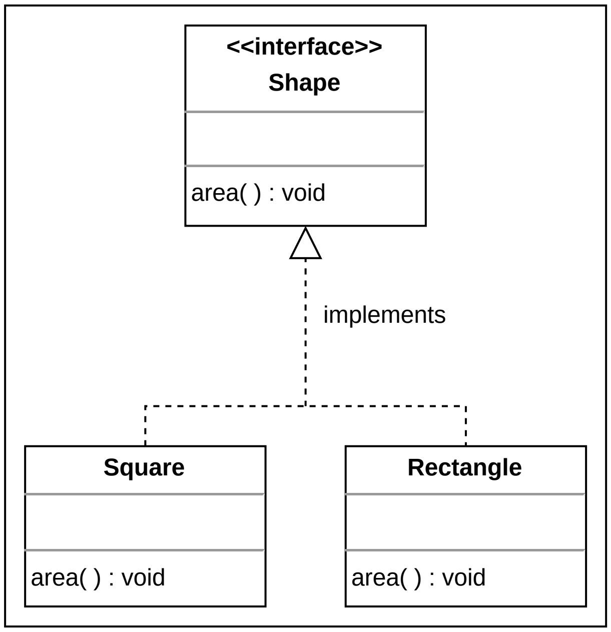
The **Interface Segregation Principle (ISP)** is a design principle that does not recommend having methods that an interface would not use and require. Therefore, it goes against having fat interfaces in classes and prefers having small interfaces with a group of methods, each serving a particular purpose.

The goal behind implementing the ISP is to have a precise code design that follows the correct abstraction guidelines and tends to be more flexible, which would help in making it more robust and reusable. This becomes key when more and more features are added to the software, making it bloated and harder to maintain.



Example

Let’s construct a simple interface called `Shape` that has the `area()` method, and `Square` and `Rectangle` as the classes to implement it as shown below:

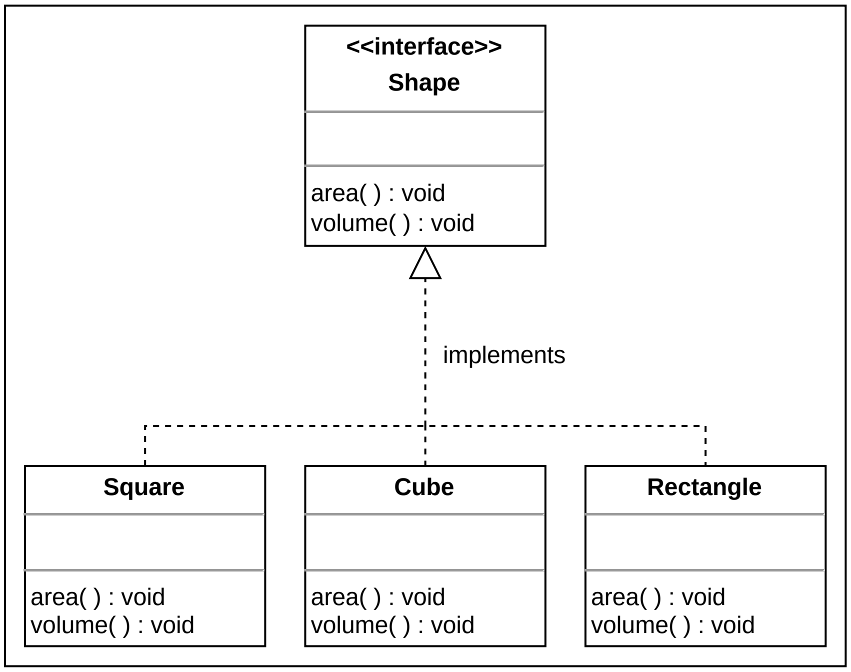


The Shape interface

So far, this implementation seems right as both the `Square` and `Rectangle` classes are implementing an interface that they’re using. Let’s see how the ISP can be violated by this example.

Violation

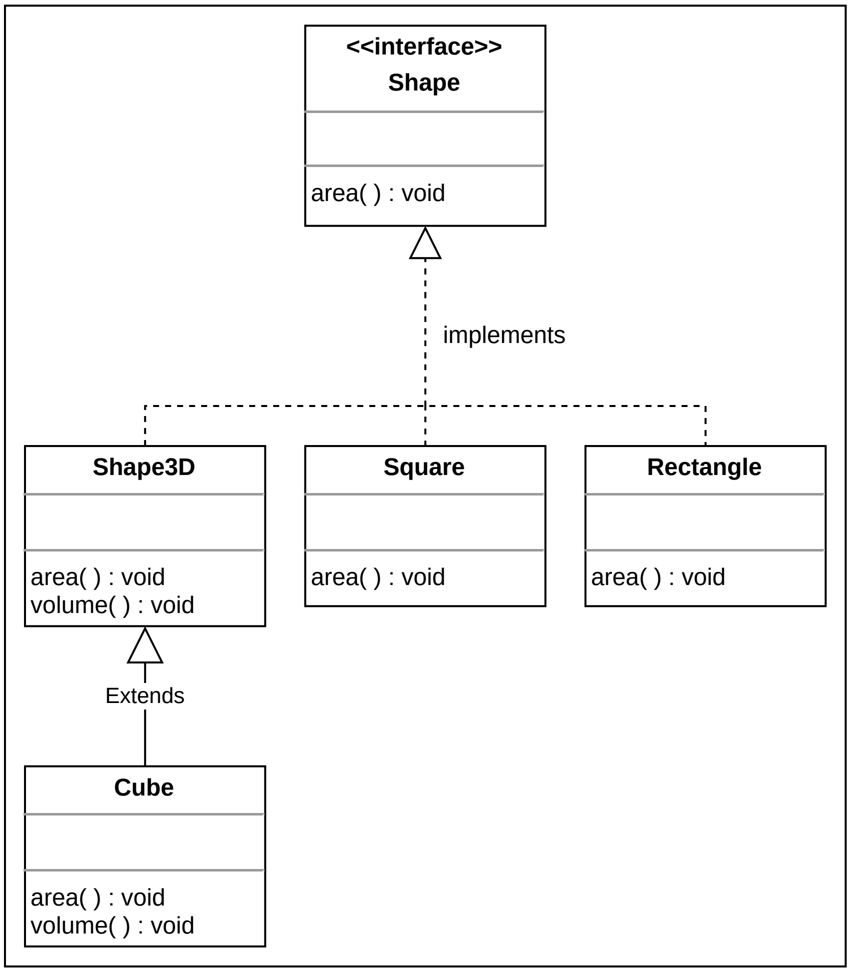
Let’s add the `volume()` method to the `Shape` interface and have a new subclass `Cube` to implement it:



Violation of the ISP

The violation leads to a problem. The 2-D shapes cannot have a volume, yet they're forced to implement the `volume()` method of the `Shape` interface that they don't have any use of. This is a clear violation of the Interface Segregation Principle.

Solution



Solution of the ISP

Now, there are two interfaces present: `Shape` and `Shape3D`. The `Shape` interface contains only the methods that are required for 2-D shapes like squares, rectangles, etc., while the `Shape3D` interface inherits the methods of the `Shape` interface and itself only contains methods for 3-D shapes like cubes, spheres, etc.

Since each class is now implementing an interface that they need to use, the Interface Segregation Principle is now no longer being violated.