

The Solid Principles of Programming

**SUBMITTED BY**

*Parth*

# 

# 

# What is solid principles?

SOLID is an acronym for the first five object-oriented design (OOD) principles by Robert C. Martin. These principles establish practices that lend to developing software with considerations for maintaining and extending as the project grows. Adopting these practices can also contribute to avoiding code smells, refactoring code, and Agile or Adaptive software development.

SOLID stands for:

* S - Single Responsibility Principle
* O - Open Closed Principle
* L - Liskov Substitution Principle
* I - Interface Segregation Principle
* D - Dependency Inversion Principle

# 

**WHY SOLID PRINCIPLES-**

1.SOLID principles are come into picture because these are the

design principles that help us in encouraging creating more

maintainable, understandable, and flexible software.

2.It also helps us in growing our application according to the

size, and can reduce the complexity of the code.

The principles in which I have worked on are:

1. Single Responsibility Principle.
2. Open Closed Principle
3. Liskov Substitution Principle.

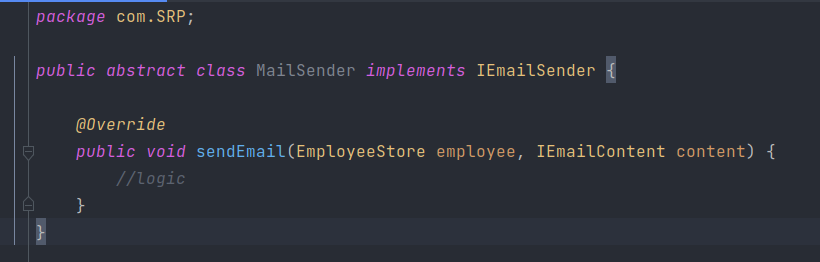


# What is the Single Responsibility Principle?

This principle states that “a class should do one thing and therefore it should have

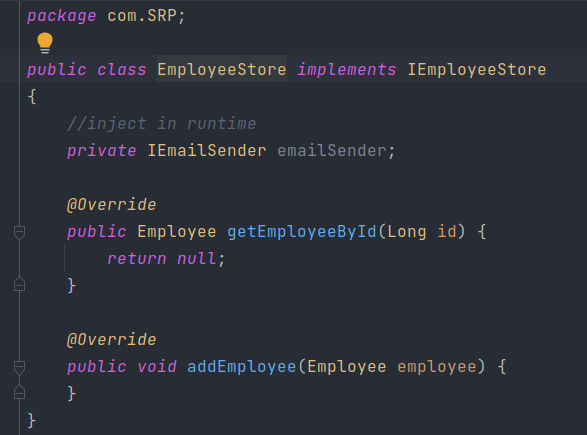
only single reason to change”. It means that every class should have a single

responsibility or single job or a single purpose.





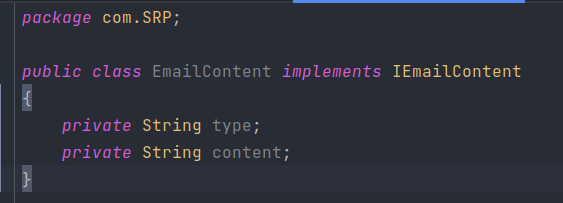
EmployeeStore class collect the data from interface IEmployeeStore as argument id



# 

# 

In EmailContent class we define the variable as private



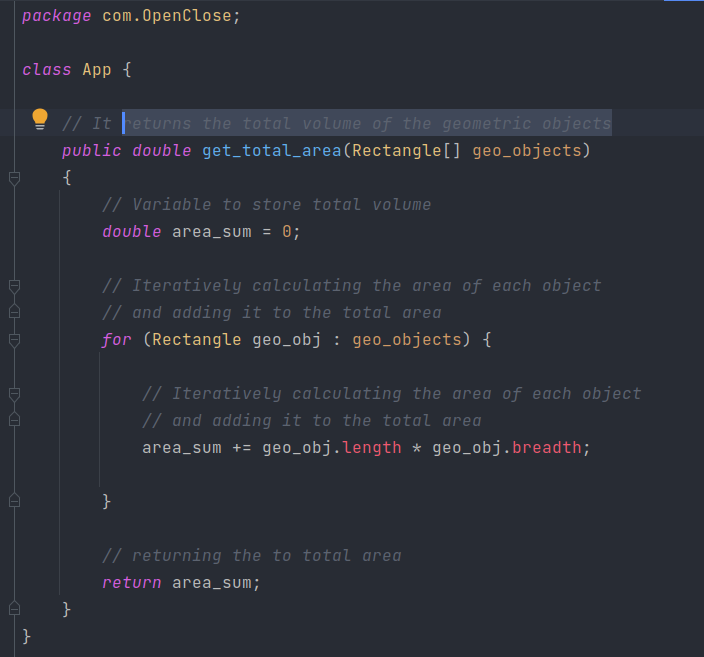
# 

1. **What is Open Closed Principles?**

This principle states that the“software entities like (classes, modules, functions and many more)

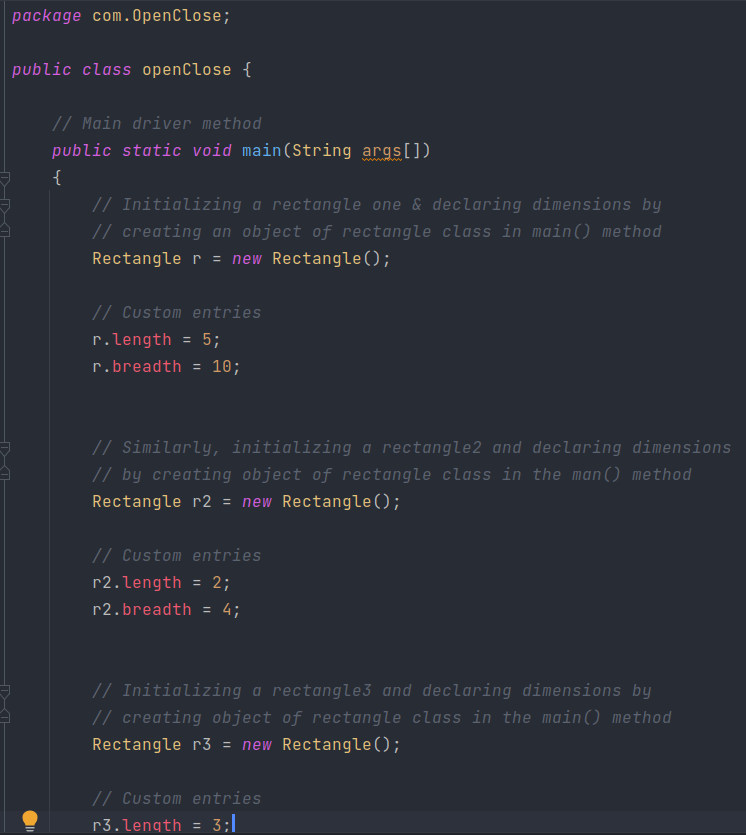
should be open for extension, but closed for modification” which means that one should

be able to extend a class behavior, with modifying it.

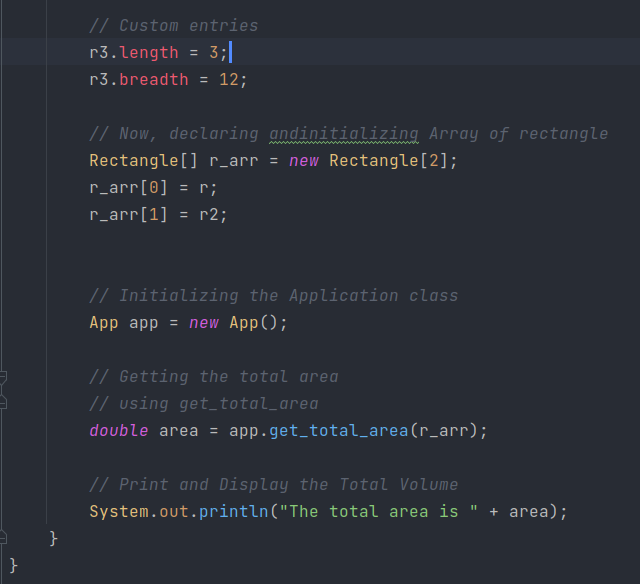




In openClose class, we have defined a main method in which we initialize the rectangle and declare the dimension of it.









1. **What is the Liskov Substitution Principle?**

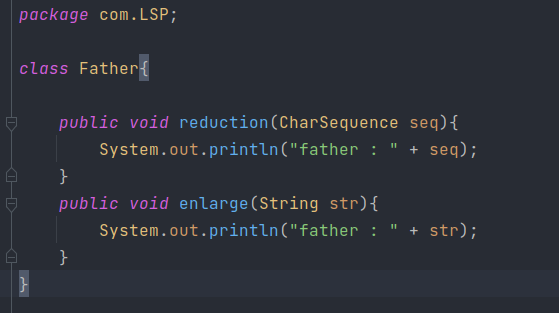
This principle states that “Derived or child classes must be substitutable for their

base or parent classes”. It was introduced by Barbara Liskov in

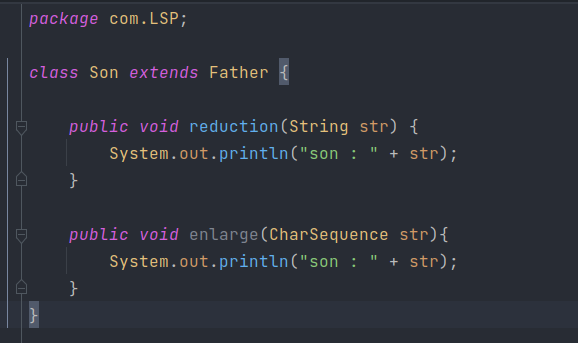
1987 and it also ensures that any class that is the child of a

parent class should be usable in place of its parent without any

unexpected behaviour.

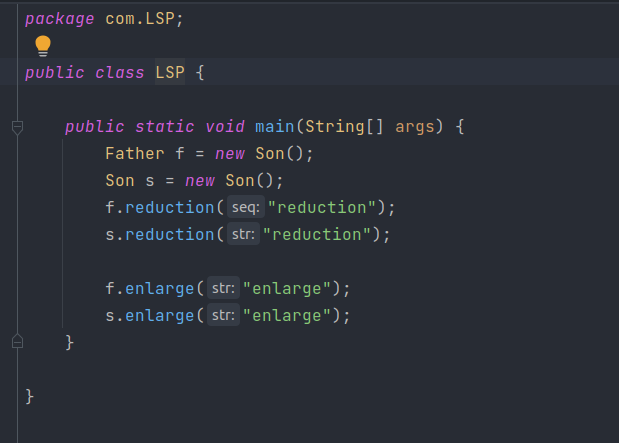








Now the main method,





**SUMMARY-**

So far, we have discussed these five principles and highlighted their

benefits. They are to help you make your code easy to adjust,

extend and test with little to no problems.

**References:**

1. [**https://www.digitalocean.com/community/conceptual\_articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design**](https://www.digitalocean.com/community/conceptual_articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design)
2. [**https://stackify.com/dependency-inversion-principle/**](https://stackify.com/dependency-inversion-principle/)