Table of Contents

Introduction	1.1
Let's Go!	1.2
S: Single Responsibility	1.3
O: Open/Closed	1.4
L: Liskov Substitution	1.5
I: Interface Segregation	1.6
D: Dependency Inversion	1.7

SOLID Design Principles In Common Lisp

Learn how to apply SOLID design principles with Common Lisp and the powerful CLOS system.

If you find any problem, want to suggest an improvement or commit changes to this book, please visit this Github repository https://github.com/common-lisp-reserve/solid-design-principles-in-common-lisp

Let's Go!

What is SOLID?

- Single Responsibility Principle
- Open/Closed Principle
- Liskov Substitution Principle
- Interface Segregation Principle
- Dependency Inversion Principle

S: Single Responsibility

A class should have one, and only one, reason to change.

Bad

```
(defclass printer ()
  ((document-type
    :initarg :document-type
    :accessor document-type)))

(defmethod process-email ((self printer))
  "process email..")

(defmethod send-email ((self printer))
  "send document as email")

(defvar printer-one (make-instance 'printer :document-type "docx"))
(process-email printer-one)
(send-email printer-one)
```

O: Open/Closed

Software entities (classes, modules, functions, etc) should be open for extension, but closed for modification.

L: Liskov Substitution

Let *Φ(x) _be a property provable about objects _x _of type _T*. Then *Φ(y) _should be true for objects _y _of type _S _where _S _is a subtype of _T.*

.

L: Interface Segregation

Clients should not be forced to depend upon interfaces that they do not use.

D: Dependency Inversion

- **High level modules should not depend upon low level modules. Both should depend upon abstractions.**
- **Abstractions should not depend upon details. Details should depend upon asbtractions.**