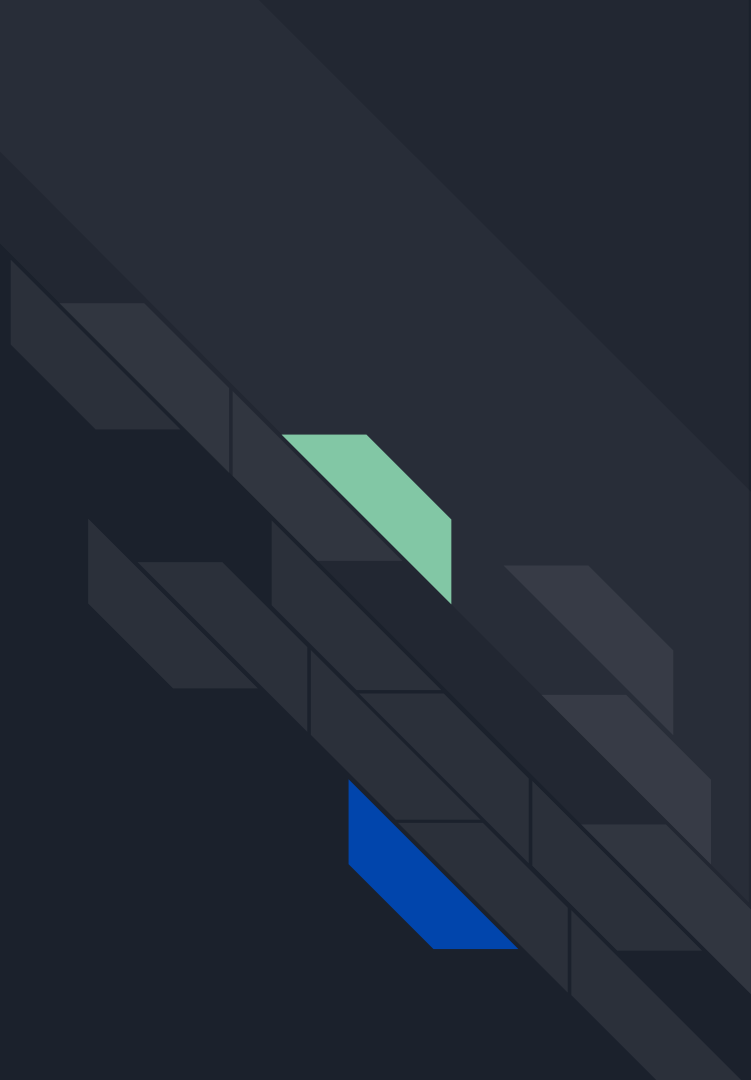
Design Principles

Mikesc Huang

SOLID



SOLID

- The Single Responsibility Principle (**S**RP)
- The Open-Closed Principle (**O**CP)
- The Liskov Substitution Principle (**L**SP)
- The Interface Segregation Principle (**I**SP)
- The Dependency Inversion Principle (**D**IP)
- 不用急，後面會介紹

Design Principles

- SOLID 告訴我們該如何把 function 和 data structure 整合到 class 裡面，還有這些 class 怎麼互相溝通
- 這裡所說的 class 不是光指 OO 裡面的 class，而是一種 function 和 data 聚集在一起的概念

Design Principles

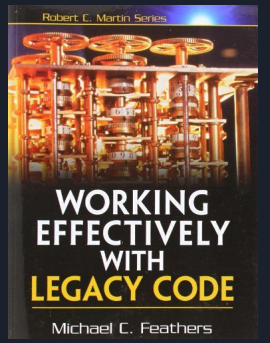
- SOLID 的目的是建立 mid-level 的軟體架構，mid-level 是指 software module
- 當然，這樣的方式可以延伸到 software component，甚至到 high-level

Design Principles

- SOLID 的歷史久遠，從 1980 年代後期作者就開始整合相關的軟體設計
- 經過時間的演進，這些設計原則有的新增，有的修改，有的刪除，有的合併
- 20 世紀初期才有一個穩定版

Design Principles

- 大約 2004 年，Michael Feathers[註1] 跟作者說可以用作者重新編寫的設計原則，取每個字首，於是 SOLID 就此誕生



註1: Michael Feathers' *Working Effectively with Legacy Code*