http://ithelp.ithome.com.tw/articles/10054846

ASP.NET 由淺入深系列 第 20 篇

91 之 ASP.NET 由淺入深 不負責講座 Day20 - ISP 介面隔離原則

前言

前面介紹了單一職責原則、開放封閉原則、最少知識原則，今天要介紹的是 Interface Segregation Principle, ISP (介面隔離原則)。

介面隔離原則，與單一職責原則和最少知識原則有一點關係，而 SOLID 原則的總綱是開放封閉原則，所以如果對這些觀念還不夠熟悉的朋友們，可以參考一下前面幾篇文章：
[ASP.NET]91 之 ASP.NET 由淺入深 不負責講座 Day17 – 單一職責原則
[ASP.NET]91 之 ASP.NET 由淺入深 不負責講座 Day18 – 開放封閉原則
[ASP.NET]91 之 ASP.NET 由淺入深 不負責講座 Day19 – LoD/LKP 最少知識原則

1. 定義

(1) Clients should not be forced to depend upon interfaces that they don't use. 也就是 Client 不應該依賴它不需要的介面。

(2) The dependency of one class to another one should depend on the smallest possible interface. 也就是 Class 之間的依賴關係應該建立在最小的 interface 上。

2. 簡單的說

Class 與外部的關係，應只依賴它需要的最小介面，所以介面不能太肥，應該要細化，不然會強迫中獎。每一個 interface 的方法數目應該盡可能地降到最低。

3. 與單一職責原則的矛盾

(1) 單一職責是指屬於同一個職責的方法應該放在一起，且一個 class/interface 只負責一個職責。可能產生一種情況，實作 interface 的 class，用不到該 interface 的所有方法。

(2) 相對於介面隔離原則，則是每一個 class 實作了某個介面，就應該代表會用到該介面所有的方法，如果不會用到所有方法，代表該 interface 可以再繼續細化。

4. 目的

對 interface 的 scope 進行約束。

5. 基本原則

(1) interface 盡量小
-粒度越小代表系統可以越靈活，但也代表結構越複雜，開發難度提升
-最小的情況就是一個 interface 只有一個 method，但違反單一職責原則
-細分 interface 的前提，就是不能違反單一職責原則

(2) interface 應高內聚
-粒度應該符合環境與未來需求中，細分到最細。
-interface 就是 contract 的觀念，對外承諾的 function 越少，變動風險越低，就越穩定 。

(3) 訂製服務
interface 只提供訪問者需要的方法 。

(4) 粒度大小的細分標準
因地因時制宜，取捨需求、開發與結構的平衡點 。

6. 結論

殺雞焉用牛刀，但書：

(1) 如果是為了殺雞，就不需要用到牛刀

(2) 如果是為了用牛刀，那就還是得用牛刀

7. 舉例

我們用下面這張 class diagram(點圖可放大)來看，可以看到以工作上所需的技能來分，一個 ExcellentWorker，應同時具備 HardSkill 與 SoftSkill。
然而，不是每一樣工作，都需要 excellent 的 worker 才能做事。一個健康的團隊，應該有將有兵，分工合作，各司其職，互助互補。

如果今天工作需要的是協助測試，那可以找 QA，也可以找 Joey。最好是先找 QA，因為如果找 Joey，他的對外方法不只測試的介面，還有許多其他的功用，就很有可能被其他任務插單。

相對的，以介面來說，一個 QA 可能只需要具備測試的介面，而不需具備 Leadership 或是 Integrity。

我自己的原則是，寧可多實作幾個 interface，保留彈性，也要避免未來因為繼承或 extend 造成強耦合而無法修改的風險。但 interface 的本身，還是要盡量符合單一職責的原則。