APPENDIX

E

UML (Unified Modeling Language) 簡介

學習目標

- 瞭解 UML 的角色與功用
- 認識 UML 各種圖表

對於軟體的開發來說，如何能夠精確地對不同的對象描述正確的軟體樣貌，是非常重要的。例如，對於需求此軟體的客戶，軟體開發者必須要能夠確認所設計的軟體的確符合客戶所需，如果等到軟體設計完成才發現與客戶的要求有出入，要再修改或是重新設計都不符經濟效益。

另外，即便是對於同屬於開發團隊的程式設計師來說，也必須要有一種方式可以讓負責的程式設計師明確的知道程式的撰寫方式，比方說資料是擺在那個資料庫中，以何種方式儲存等等。如果缺乏這種明確溝通的方式，同一團隊的不同程式設計師各自爲政，結果可能是每一位程式設計師都寫好了程式，但是卻無法組合成完整的系統。

UML 就是一種透過不同觀點描述軟體系統的語言，它採用各式各樣的圖表作爲呈現的方式。如果把軟體開發比喻爲蓋房子，那麼建築物的模型以及建築物藍圖等等就等同於 UML 的角色，這些都是在實際建構之前描述完成品的工具。透過建築物的模型，客戶或是建商可以在開工之前瞭解建築物的外觀；而透過建築藍圖，建築工人就可以知道如何建構這棟房子，如果缺乏這些工具，要想蓋出正確、安全的房子，那可就難上加難了。

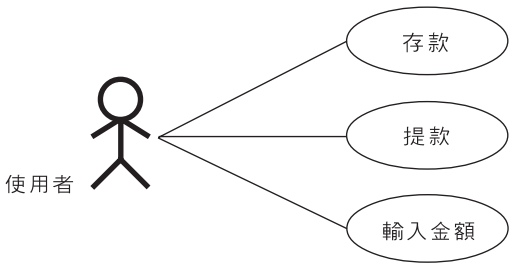
對於 UML 來說，它會以下列各種圖表描述所要建構的軟體系統，每一種圖表代表一種觀點，分別描述了軟體的某些意涵：

- 使用案例圖 (Use Case Diagram)
- 類別圖 (Class Diagram)
- 物件圖 (Object Diagram)
- 順序圖 (Sequence Diagram)
- 合作圖 (Collaboration Diagram)
- 狀態圖 (State Diagram)
- 活動圖 (Activity Diagram)
- 元件圖 (Component Diagram)
- 部署圖 (Deployment Diagram)

以下我們就針對其中比較常用也比較重要的圖表簡單介紹其意義與畫法，期望能讓您對於 UML 有基本的認識，如果需要進一步瞭解 UML，可以閱讀其他的專書。

E-1 使用案例圖

使用案例圖顧名思義，描述的是所建構的軟體系統各種可能的使用情境，一般而言，這就是系統使用者所會看到的各項功能。舉例來說，以下就是 14-36 頁範例程式 TestAccount.java 的使用案例圖：

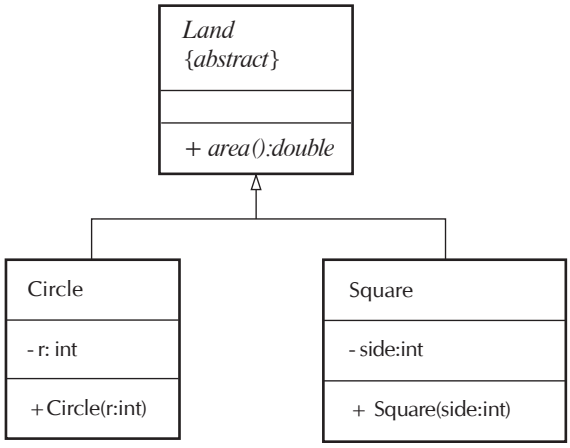


這個圖就表示了會使用範例程式的是使用者，在 UML 中稱其爲**動作者 (Actor)**。使用者使用這個系統時，可進行的動作 (也就是『**使用案例**』) 包括：選擇存款或提款，以及輸入存提款金額等三種。因此，當我們實際撰寫程式時，就必須提供對應於這三種使用案例的功能。

一般而言，使用案例圖很適合運用在與系統需求者的溝通上，以確認所有需求都會出現在未來建構出的軟體中，並且符合使用者的實際情境。

E-2 類別圖

在第 8 章曾經提過，使用物件導向的方式來建構軟體時，最重要的就是要找出需要哪些角色，也就是類別。UML 中的類別圖就是要用來描述這些角色，將每一個類別的特性與行爲列出，並且表達出類別之間的關係。舉例來說, 以下就是書中 12-4 頁 AbstractLand.java 的類別圖：



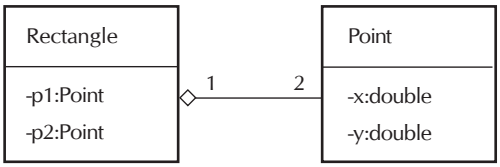
在類別圖中, 每一個矩形就是一個類別。矩形中由上而下分別列出類別名稱、成員變數、成員函式。以上圖爲例, Circle 類別有成員變數 r, 其資料型別爲 int；此外它也有一個建構方法, 需使用 1 個 int 型別的參數。各成員前可加上下列符號表示其存取控制：

- -：Private
- +：Public
- #：Protected

透過這個方式, 就可以清楚的看出每個類別的屬性以及行爲。

此外 Land 類別名稱下標註的 {abstract}, 表示它是個抽象類別, 有時也會用斜體字來書寫抽象類別及抽象方法的名稱, 如圖中的 area() 方法。而 Circle 、 Square 類別都以一條實線及空心箭頭指向 Land 類別, 表示它們都是 Land 的子類別。

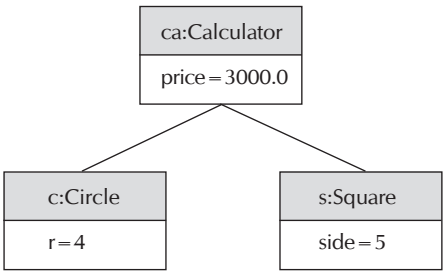
除了標示類別的內容以及繼承關係外，當類別彼此具有包含的關係時，也可用類別圖表現出來。舉例來說，假設代表矩形的類別 Rectangle，它有兩個代表左上與右下座標的成員變數 p1 、p2，而它們則是座標點 Point 類別的物件，此時 Rectangle 類別與 Point 類別之間的關係可畫成：



這種關係稱爲組合 (Composition)，而線條邊的數字則代表『1』個 Rectangle 物件含有『2』個 Point 物件。

E-3 物件圖

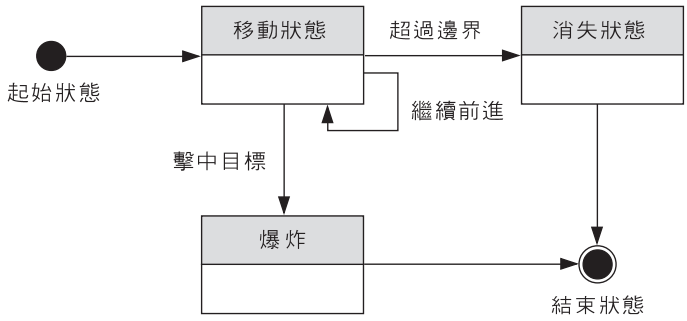
物件圖表示的是某個時間點特定的一組物件之間的關連，相同的資訊雖然也可以透過類別圖來展現，但是當物件之間的關係比較複雜時，就可以使用物件圖來展現。以下就是 AbstractLand.java 中當主程式建立了所有的物件之後間的物件圖：



圖中就表示了 ca 這個 Calculator 物件會和 Circle 物件 c 、Square 物件 s 有關連。

E-4 狀態圖

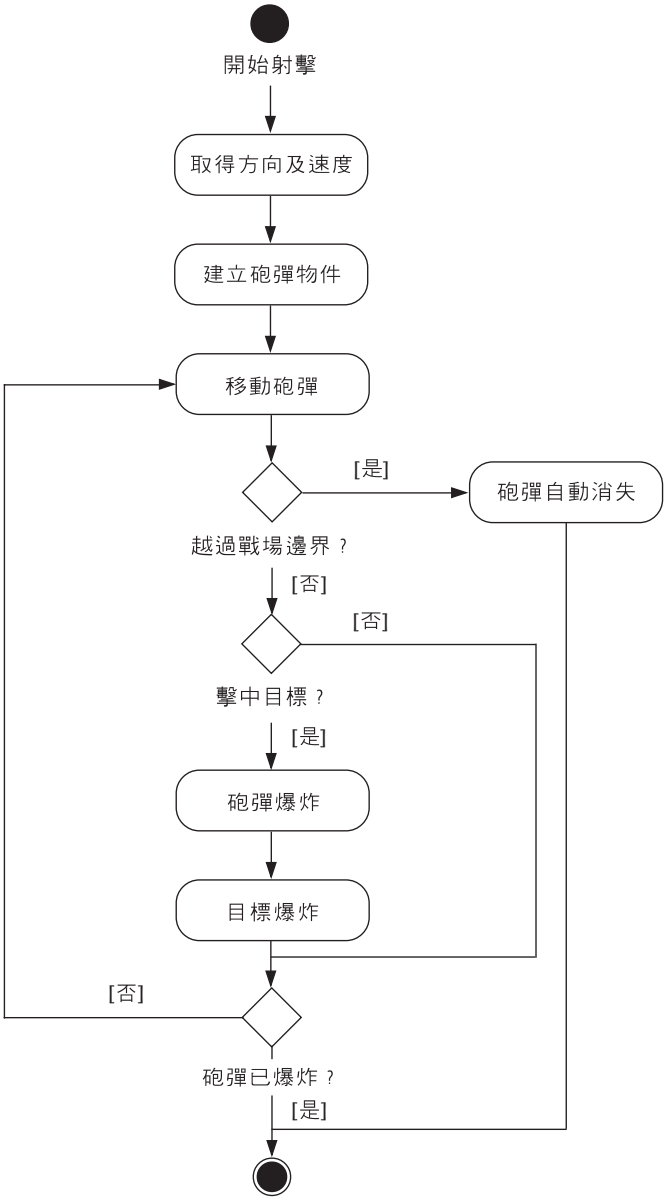
狀態圖是用來表現物件在不同情境下的狀態，例如假設有個射擊遊戲程式，程式中砲彈物件的狀態就可用下圖來表示：



一般來說，如果物件的狀態非常複雜，就很適合使用狀態圖來展現。

E-5 活動圖

活動圖所展現的就是程式的運作邏輯，也就是流程，這對於釐清特定的使用情境進行的細節很有幫助。例如以下就是射擊遊戲中射擊情境的活動圖：



在這個圖中，可以很清楚的看到，射擊的情境是由玩家輸入代表砲彈速度及方向的向量開始。接著移動砲彈，並檢查是否越過戰場邊界，然後檢查是否擊中武器庫。最後，如果砲彈已經越過戰場消失或是擊中目標而爆炸，就結束射擊情境，否則就重複移動砲彈與檢查的動作。