

# 第 9 章

## 尋找類別方法

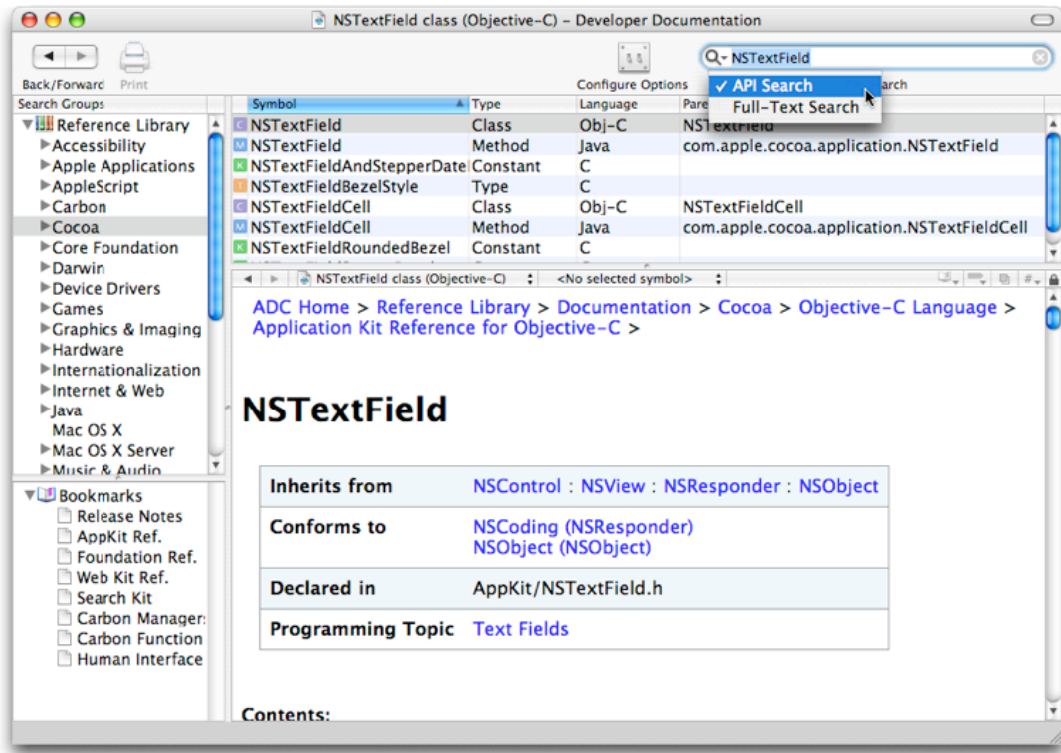
在前一章中，我們學到了方法這個東西。我們自己撰寫了兩個方法（的本體），但是我們也用了一個 Apple 提供的。 `setIntValue:` 是在文字欄位物件中顯示數值的方法。但是您要如何找到這個可以用的方法呢？

請記住，對於您用的每一個由 Apple 創造的方法，您都不需要再自己額外撰寫任何程式碼。此外，它可能也會更沒有臭蟲（bug-free）。因此，在設計您自己的方法之前先花點時間查查看是不是已經有了可用而適合的方法是很值得的。

在 Interface Builder 中，如果您把游標停留在面板視窗中的物件上，會有一個小標籤跳出來。如果您把游標停留在按鈕圖示上，您會看到 "NSButton"。如果您停留在顯示為 "System Font Text" 的文字欄位上，您會看到 "NSTextField"。這些每一個都是一個類別名稱。讓我們來看看 NSTextField 類別有什麼可用的方法。

進入 Xcode，在選單上選取 *Help->Documentation*。在左邊的頁框中，選取 *Cocoa* 然後在搜尋欄位輸入 "NSTextField"（請確定 API-Search 模式是選取起來的，參閱下面的螢幕截圖）。在您一邊輸入的時候，清單中的候選結果就會一邊減少，很快的您就會看到 NSTextField 出現在最上面。

點擊 NSTextField 所在的那行（類別為 *Class*）來讓 NSTextField 類別的資訊顯示在下方的頁框中。



用 Xcode 在 Cocoa 說明文件中導覽。

首先您應該注意到的是這個類別是繼承自其他一系列的類別。在列表最後面的是最頂點，NSObject。下面一點（請往下捲）是標題：

## Method Types

這就是我們要開始尋找的地方。快速的瀏覽一下，會發現我們在這裡找不到我們所需要的方法以用來在文字欄位物件上顯示數值。因為繼承的原則，我們需要查看 NSTextField 類別的直屬父類別，NSControl（而如果我們又失敗了，我們必須繼續檢查它的父類別 NSView，以此類推）。因為所有的文件都是 HTML 格式，所以我們只需要點下顯示在 *Inherits from* 清單中的文字 NSControl，就可以顯示出 NSControl 類別的資訊：

## NSControl

**Inherits from**      **NSView : NSResponder : NSObject**

您可以看到我們上移了一個類別。在方法清單中，我們注意到了一個子標題：

## Setting the control's value

那正是我們要的，用來設定一個數值。在這個子標題下面我們找到：

– setValue:

聽起來很合理，所以我們來查看一下它的敘述，請點擊 *setIntValue*: [連結](#)。

`setIntValue:`

- (void)setIntValue:(int)anInt

Sets the value of the receiver's cell (or selected cell) to the integer `anInt`. If the cell is being edited, it aborts all editing before setting the value; if the cell doesn't inherit from `NSActionCell`, it marks the cell's interior as needing to be redisplayed (`NSActionCell` performs its own updating of cells).

在我們的程式中，我們的 `NSTextField` 物件是接收者，而我們需要餵給它一個整數。我們也可以由這個方法的標記式（signature）看出來：

在 Objective-C 中，負號標示了一個實體方法宣告的開始（相對的是類別方法，我們稍後會談到）。`void` 表示不會傳回任何東西給方法的喚起者。意思是，當我們傳送一個 *setIntValue*: 訊息給 `textField`，我們的 `MAFood` 物件不會由文字欄位物件那接收到任何的值得傳回來。這沒什麼關係。在冒號後面，`(int)` 表示變數 `anInt` 必須是一個整數。在我們的例子中，我們傳送給它的是數值 5 或 0，而它們是整數沒錯，因此我們不會遇到任何問題。

有時候要找到適當的方法來用是有點困難的，但是當您更熟悉這份文件之後情況會比較好的，所以，請保持練習。

如果您要從文字欄位物件 `textField` 中讀取數值又該如何呢？還記得函式最棒的地方就是所有它內部的變數都被隔離起來了嗎？方法也一樣。通常物件會有一對相關的方法，稱為 "存取函式"（Accessors），一個用來讀取數值，一個用來設定數值。我們已經知道了後者，也就是 *setIntValue*: 方法，前者看起來則像下面這樣：

```
[1]
- (int) intValue
```

如您所見，這個方法回傳一個整數。因此，如果您要讀取我們 `textfield` 物件的整數數值，我們要傳送一個如下的訊息給它：

```
[2]
resultReceived = [textField intValue];
```

再一次，在函式（與方法）中，所有的變數都被隔離開了。這對於變數名稱的選擇與使用非常好，因為您不需要害怕在您程式中某個部份設定變數的這個動作會影響到在別處函式中擁有相同名稱的另一個變數。然而，函式名稱仍然必須是唯一的。Objective-C 進行了更進一步的隔離：方法名稱只需要在同一個類別內是唯一的即可，而不同的類別間

則可以有相同的方法名稱。這對於大型程式是一個很好的功能，因為程式設計師們可以各自撰寫類別，而不用擔心彼此的方法名稱產生衝突。

還有呢。在不同類別中的不同方法可以有相同的名稱，這在怪胎術語中被成為 "多型" (polymorphism)，這是讓物件導向程式設計 (object-oriented programming) 如此與眾不同的其中一個原因。它讓您可以在不需要事先了解您在使用的物件的類別的情況下就能夠撰寫一大堆的程式碼。所有的一切只需要在執行期間 (run-time) 時，這個真正的物件能夠了解您傳送給它的訊息，如此而已。善用這個好處，您就可以撰寫在設計上有彈性又有擴充性的程式。舉例來說，在我們創造的 GUI 程式中，如果我們用其他能夠了解 *setInValue*: 訊息的類別的物件來取代文字欄位，我們的程式仍然可以在不修改程式碼甚至不重新編譯的情況下正常運作，甚至可以在執行期間變更物件而不中斷任何事情。這一點所依靠的就是物件導向程式設計的威力。