07 CHAPTER

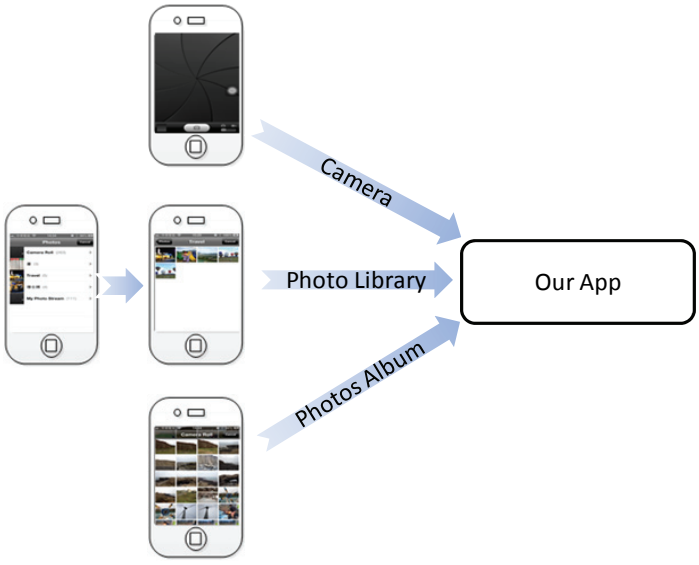
拍照與音樂

拍照與播放音樂可算是智慧型手機在日常生活中最常被使用的兩項功能。以 iPhone 為例，800 萬畫素的鏡頭使得它的拍照效果並不輸一般的數位相機，再加上輕巧及攜帶方便的特性，它已逐漸成為智慧型手機持有者最常使用的拍照工具。隨之而生的還有越來越多運用到拍照功能的 App。所以本章的任務之一便是討論如何將拍照及錄影功能加入 App 中。

事實上，我們得先感謝 iOS 內建的相機服務，因為它既可以拍照也可以錄影，所以我們的 App 若需要使用拍照或是錄影功能，只要直接呼叫相機服務即可。既然有相機服務可以讓我們呼叫，當然相關的介面設計也不用煩惱，因為這個部份 iOS SDK 也都已經幫我們準備好了，我們需要做的就是呼叫服務而已。最後，拍完照、錄完影，內建的相機服務會將資料傳回給呼叫的 App 進行後續的處理，例如存檔。呼叫相機服務後所看到的介面設計與蘋果公司提供的「相機 App」介面是相同的， 除非您想要寫一個專門用來拍照或錄影的 App，而將介面大規模的重新設計，不然，系統提供的介面已經足夠一般使用。

另外，對一個需要照片做進一步處理的 App 而言，相機只是提供照片來源的管道之一。一個需要處理照片的 App，其照片管道來源總共有三個，除了相機之外，還有「照片圖庫（Photo Library）」與「相簿（Photos Album）」。照片圖庫與相簿都是從已經存檔的照片中挑選一張來處理，跟相機的處理邏輯幾乎一樣，只不過相機是即時拍一張照片，而其他兩個都是從存檔的照片中挑選。照片圖庫與相簿的差異在於，照片圖庫會先呈現照片目錄給使用者看，使用者

可以先選擇目錄，爾後再從選擇好的目錄中選取照片，而相簿則是直接從 Camera Roll 這個相簿中挑選照片，Camera Roll 是系統預設的相簿，裡面存放裝置內所有的照片。順道一提，雖然一張照片可以屬於很多的相簿，但必定會屬於 Camera Roll 這個相簿而且也不能被移除。



本章另一部份與音樂的播放有關。從對日常生活的觀察中可以發現，大多數使用者走在路上或是搭公車的時候，都會掛個耳機聽音樂，所以播放音樂是行動裝置的另一項重要功能。除了平常聽音樂之外，玩遊戲的時候也需要播放特殊的音效或音樂來讓遊戲玩起來更有樂趣。

想要播放原本已經存在於 iPhone 或 iPad 內的音樂，程式碼寫起來是很簡單的，其處理邏輯跟選取一張已經存在的照片差不多。也就是說，iOS 已經幫我們設計好了音樂清單介面，我們的程式只要呼叫這個介面，使用者就可以選取音樂，選取完後畫面會再返回我們的 App，此時 App 只要根據使用者選取的音樂去播放即可。播放方式可以分為兩種：一種稱為 application 播放，另一種為 iPod 播放。這裡的 iPod 不要誤會只有 iPod 能使用，那只是一種播放方式的名稱而已。Application 模式是播放音樂的控制都是由我們寫的 App 來處理，當 App 進入到背景執行時，音樂就會停止。iPod 模式則是將音樂播放交由系統內建的播放程式去播放，此時我們的 App 進入背景後音樂都不會停止，因為播放音樂並不是我們的 App 而是 iOS 系統。

7-1 讓手機震動

難易度：★

手機震動其實不算是播放音樂，只是 iOS SDK 將震動變成播放音樂的一個參數，所以我們利用用播放音樂的方式來播放「震動」即可。

適用版本：iOS 6 / Xcode 4.5
Framework：AudioToolbox.framework

步驟與說明

1 建立 Single View Application 專案，並且載入 AudioToolbox.framework。

2 在 AppDelegate.h 中匯入 AudioToolbox.h 標頭檔。

```
#import <UIKit/UIKit.h>
#import <AudioToolbox/AudioToolbox.h>

@interface AppDelegate : UIResponder <UIApplicationDelegate>

@property (strong, nonatomic) UIWindow *window;

@end
```

3 開啟 AppDelegate.m，在 applicationDidBecomeActive: 方法中處理手機的震動，這個方法是當 App 進入 Active 狀態後就會觸發，所以當 App 以 HOME 鍵進入背景執行模式後再重新執行時便會觸發這個方法。

```
- (void)applicationDidBecomeActive:(UIApplication *)application
{
    AudioServicesPlaySystemSound(kSystemSoundID_Vibrate);
}
```

4 執行看看。執行後按 HOME 鍵，然後再點選這個 App 的 icon 讓他恢復到前景執行。

7-4 從相本中挑選一張照片（iPhone）

難易度：★

從 iPhone 中已經存在的相本挑選其中一張照片。

適用裝置：iPhone
先備知識：7-3 開啟相機拍照並存檔
Framework：AssetsLibrary.framwork、MobileCoreServices.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 AssetsLibrary.framework 與 MobileCoreServices.framework 加到專案中。

3 開啟 Storyboard，在 View Controller 畫面上加一個按鈕元件和一個 Image View 元件。使用者按下這個按鈕後可以開啟相簿，並且挑一張照片後顯示在 Image View 元件中。這裡我們先將 Image View 元件的圖片顯示方式改為「Aspect Fit」，讓照片可以 fit 到 Image View 元件一樣的大小。

4 開啟 ViewController.h，這個類別需符合 UINavigationControllerDelegate 與 UIImagePickerControllerDelegate 這兩個協定的規範。除此之外，設定 Image View 元件的 IBOutlet 變數 myImg。

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController
      <UINavigationControllerDelegate, UIImagePickerControllerDelegate>
@property (weak, nonatomic) IBOutlet UIImageView *myImg;

@end
```

5 在 View Controller.m 中設定按鈕的 IBAction 方法，在這個方法中開啟相片瀏覽畫面。

```
- (IBAction)buttonPress:(id)sender
{
    UIImagePickerController *imagePicker = [[UIImagePickerController
                                                        alloc] init];
    // 設定相片來源來自於手機內的相本
    imagePicker.sourceType =
                            UIImagePickerControllerSourceTypePhotoLibrary;
    imagePicker.delegate = self;
    // 開啓相片瀏覽界面
    [self presentViewController:imagePicker animated:YES completion:nil];
}
```

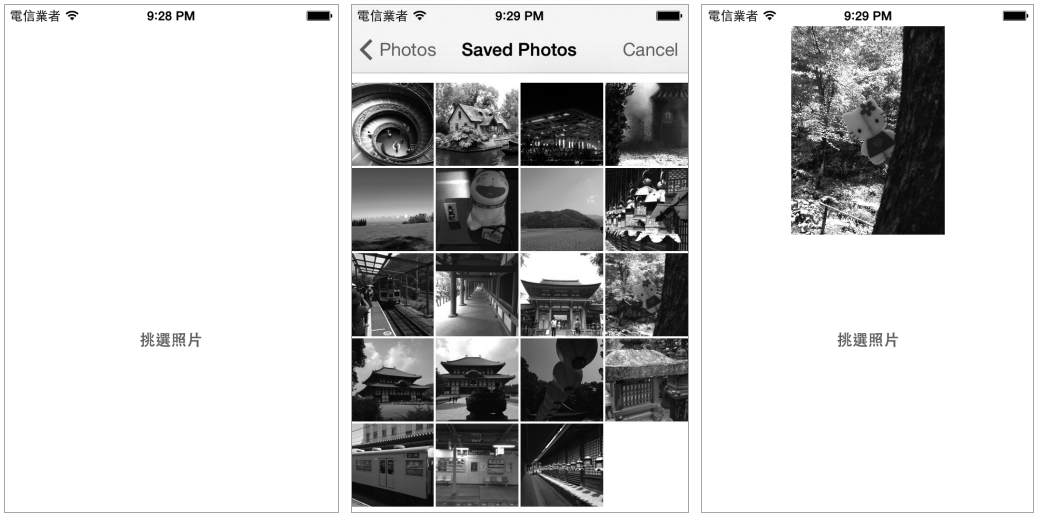
6 相片瀏覽畫面開啟後，使用者可以從中挑選一張照片或是取消。如果要取得使用者挑選的照片，就必須在 ViewController.m 中實作 imagePickerController: didFinishPickingMediaWithInfo: 方法。

```
-(void)imagePickerController:(UIImagePickerController *)picker
                      didFinishPickingMediaWithInfo:(NSDictionary *)info
{
    // 取得相片
    UIImage *image = [info
                       valueForKey:UIImagePickerControllerOriginalImage];
    // 放到 Image View 元件上
    self.myImg.image = image;
    // 關閉相片瀏覽界面
    [self dismissViewControllerAnimated:YES completion:nil];
}
```

7 如果使用者按下取消按鈕，就必須在 ViewController.m 中實作 imagePickerControllerDidCancel: 方法，在這個方法中關閉照片瀏覽畫面。

```
-(void)imagePickerControllerDidCancel:(UIImagePickerController *)picker
{
    // 使用者按下取消扭後關閉相片瀏覽界面
    [self dismissViewControllerAnimated:YES completion:nil];
}
```

8 執行看看。



7-5 從相本中挑選一張照片（iPad）

難易度：★

由於 iPad 螢幕比 iPhone 來得大，因此，開啟照片瀏覽畫面讓使用者去選擇一張照片的方式跟 iPhone 不一樣。差異在 iPhone 的照片瀏覽畫面是佔滿整個螢幕，但是 iPad 為了整體介面的美觀，所以要先開啟一個小視窗，稱為 popover 視窗，然後再將照片瀏覽畫面放到那個小視窗中讓使用者挑選。iOS 強制 iPad 的程式設計師必需要這樣做，否則 Xcode 的編譯過程會出現錯誤訊息。

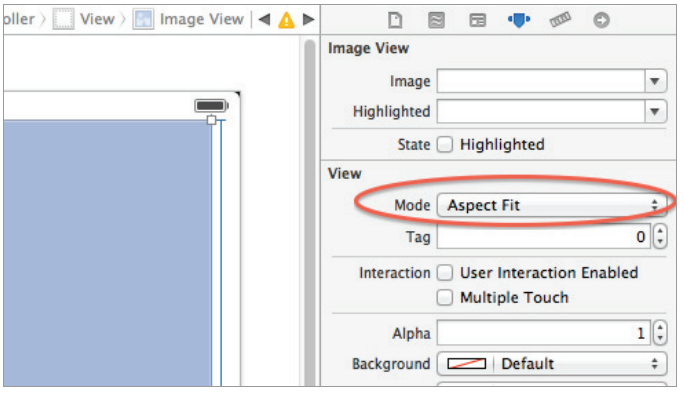
適用裝置：iPad
先備知識：7-3 開啟相機拍照並存檔、7-4 從相本中挑選一張照片（iPhone）
Framework：AssetsLibrary.framwork、MobileCoreServices.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 AssetsLibrary.framework 與 MobileCoreServices.framework 加到專案中。

3 開啟 Storyboard，在 View Controller 頁面上加上一個按鈕元件和一個 Image View 元件。我們打算這個按鈕按下去後開啟相簿，並且挑一張照片後放到 Image View 元件中。Image View 元件的圖片顯示方式改為「Aspect Fit」。



4 開啟 ViewController.h，這個類別需要符合 UINavigationControllerDelegate、UIImagePickerControllerDelegate 與 UIPopoverControllerDelegate 這三個協定的規範。除此之外，再宣告一個 UIPopoverController 的實體變數，這個變數要用來開啟 popover 視窗。

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController
     <UINavigationControllerDelegate, UIImagePickerControllerDelegate,
     UIPopoverControllerDelegate>
{
    UIPopoverController *popover;
}
@property (weak, nonatomic) IBOutlet UIImageView *myImg;

@end
```

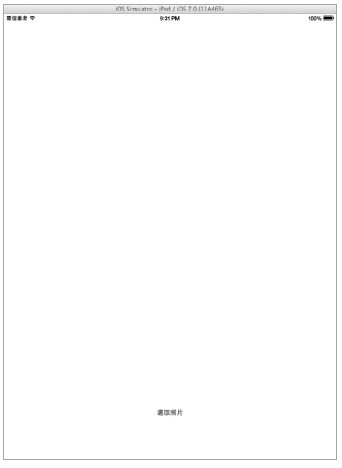
5 在 View Controller.m 中設定按鈕的 IBAction 方法，在這個方法中使用 popover 視窗開啟相片瀏覽畫面。當 popover 視窗要準備開起來時，其中有個參數 UIPopoverArrowDirectionAny 的意思是 popover 視窗上的小箭頭指向的方像是由系統決定的，除此之外 popover 視窗出現的位置也是系統根據當時的狀況自動以最好的方式來呈現。

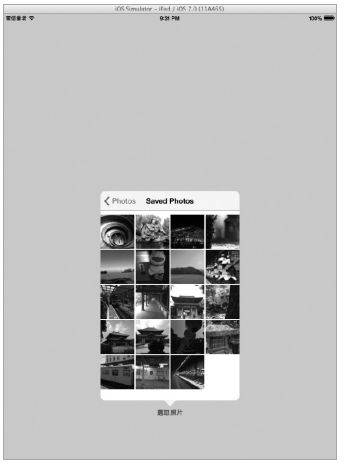
```
- (IBAction)buttonPress:(UIButton *)sender
{
    UIImagePickerController *imagePicker = [[UIImagePickerController
                                            alloc] init];
    // 設定相片的來源為手機內的相本
    imagePicker.sourceType =
                          UIImagePickerControllerSourceTypePhotoLibrary;
    imagePicker.delegate = self;

    // 初始化 popover 視窗，其內容為相片導覽畫面
    popover = [[UIPopoverController alloc]
                                initWithContentViewController:imagePicker];
    // 取得按鈕的座標與長寬大小
    CGRect rect = sender.frame;
    // 在按鈕周圍開啟 popover 視窗
    [popover presentPopoverFromRect:rect inView:self.view
      permittedArrowDirections:UIPopoverArrowDirectionAny animated:YES];
}
```

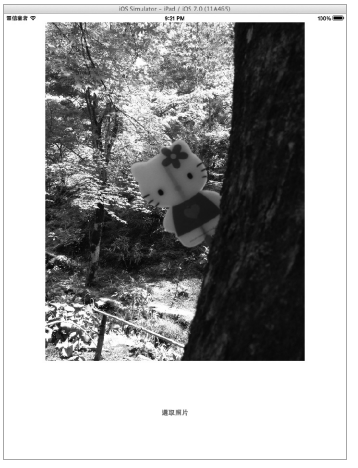
6 Popover 畫面開啟後，使用者可以從中挑選一張照片或是點選 popover 視窗以外的地方取消選取。如果要取得使用者挑選的照片，就必須在 ViewController.m 中實作 imagePickerController:didFinishPickingMediaWithInfo:方法。如果是取消選取，iOS 會自動關掉 popover 視窗，因此就不用像 iPhone 版一樣還需要實作 imagePickerControllerDidCancel: 方法。

```
-(void)imagePickerController:(UIImagePickerController *)picker
                    didFinishPickingMediaWithInfo:(NSDictionary *)info
{
    // 取得相片
    UIImage *image = [info
                      valueForKey:UIImagePickerControllerOriginalImage];
    // 放到 Image View 元件上
    self.myImg.image = image;
    // 關閉 popover 視窗
    [popover dismissPopoverAnimated:YES];
}
```

7 執行看看。







7-6 一次取得相本中所有照片

難易度：★★

如果想要寫一個類似 Photos Gallery 之類的 App，我們就必須一次取得相本中的所有照片後展示，而不是讓使用者逐一挑選照片。

適用裝置：iPhone / iPad

Framework：AssetsLibrary.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 AssetsLibrary.framework 加到專案中。

3 開啟 ViewController.h，先匯入 AssetsLibrary.h 標頭檔，然後宣告兩個變數，library 負責讀取照片，imageArray 則用來儲存取出的照片。

```
#import <UIKit/UIKit.h>
#import <AssetsLibrary/AssetsLibrary.h>

@interface ViewController : UIViewController
{
    ALAssetsLibrary *library;
    NSMutableArray *imageArray;
}
@end
```

4 開啟 ViewController.m，在 viewDidLoad 方法中讀取照片。取出照片的過程是使用非同步的方式，也就是 iOS 會使用另外一個執行續讀取照片。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    library = [[ALAssetsLibrary alloc] init];

    // 使用參數 ALAssetsGroupSavedPhotos 取出所有存檔照片
    [library enumerateGroupsWithTypes:ALAssetsGroupSavedPhotos
                        usingBlock:^(ALAssetsGroup *group, BOOL *stop) {
        NSMutableArray *tempArray = [[NSMutableArray alloc] init];
        if (group != nil) {
```

```
            [group enumerateAssetsUsingBlock:^(ALAsset *result,
                                               NSUInteger index, BOOL *stop) {
                if (result != nil) {
                    [tempArray addObject:result];
                }
            }];
            // 保存結果
            imageArray = [tempArray copy];
            NSLog(@"取出照片共 %d 張", [imageArray count]);
        }
    } failureBlock:^(NSError *error) {
        // 讀取照片失敗
    }];
}
```

5 完成。

```
GetAllPhotos[7263:907] 取出照片共 271 張
```

7-7 播放 App 內建的音樂

難易度：★

播放專案中的音樂檔，注意此音樂檔並非已經存在於行動裝置內的音樂，而是跟隨專案的，也就是說如果使用者刪除此 App，音樂就跟著刪除。

適用裝置：iPhone / iPad

Framework：AVFoundation.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 AVFoundation.framework 加到專案中，並且在專案中加入一個 music.mp3 的音樂檔。

3 開啟 ViewController.h，先匯入 AVFoundation.h 標頭檔以及讓這個類別符合 AVAudioPlayerDelegate 協定的規範，除此之外，再宣告一個 AVAudioPlayer 型態的變數 audioPlayer 來播放音樂。

```
#import <UIKit/UIKit.h>
#import <AVFoundation/AVFoundation.h>

@interface ViewController : UIViewController <AVAudioPlayerDelegate>
{
    AVAudioPlayer *audioPlayer;
}
@end
```

4 開啟 ViewController.m，在 viewDidLoad 方法中先初始化 audioPlayer 變數，並設定代理人類別為 ViewController，這個目的是當音樂播放過程中被某些因素中斷時（例如電話鈴響），程式可以回應這個中斷狀況，並且當中斷因素結束後，我們可以再恢復音樂播放。這些都準備就緒後，就可以播放音樂了。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    // 找到 music.mp3 的路徑
    NSString *filePath = [[NSBundle mainBundle] pathForResource:@"music"
                          ofType:@"mp3"];
    NSData *fileData = [NSData dataWithContentsOfFile:filePath];

    // 初始化 audioPlayer
    audioPlayer = [[AVAudioPlayer alloc] initWithData:fileData
                   error:nil];
    // 設定代理人類別
    audioPlayer.delegate = self;

    // 檢查是否一切就緒，如果沒問題就播放音樂
    if (audioPlayer != nil) {
        if ([audioPlayer prepareToPlay])
            [audioPlayer play];
    }
}
```

5 實作 audioPlayerBeginInterruption: 方法，這個方法是當音樂被中斷時呼叫。

```
-(void)audioPlayerBeginInterruption:(AVAudioPlayer *)player
{
    // 音樂被系統中斷，例如有電話進來
    NSLog(@"音樂中斷");
}
```

6 實作 audioPlayerEndInterruption:withOptions: 方法，這個方法是當音樂中斷因素結束後呼叫，我們可以在這個方法中恢復音樂播放。

```
-(void)audioPlayerEndInterruption:(AVAudioPlayer *)player
                                        withOptions:(NSUInteger)flags
{
    // 音樂中斷因素結束，從 flags 參數判斷音樂中斷時的狀態
    NSLog(@"音樂中斷結束");
    if (flags == AVAudioSessionInterruptionOptionShouldResume)
        [player play];
}
```

7 執行看看。

7-8 顯示目前音樂播放進度

難易度：★★

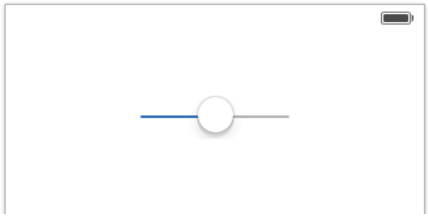
在播放音樂的時候，可以利用一個長條圖來顯示目前音樂播放進度並且可以讓音樂直接跳到某個特定的時間去播放。

適用裝置：iPhone / iPad
先備知識：7-7 播放 App 內建的音樂、15-7 計時器
Framework：AVFoundation.framework

步驟與說明

1 根據先備知識 1 建立 Single View Application 專案。

2 開啟 Storyboard，我們從元件庫中拖放一個 Slider 到 View Controller 上。在這裡我們不用 Progress View 元件的原因是 Slider 元件上有個指示器，我們要讓使用者可以移動這個指示器來改變音樂的播放位置。



3 開啟 ViewController.h，除了根據先備知識 1 所宣告的 audioPlayer 變數外，另外再根據先備知識 2 宣告一個 NSTimer 類別的實體變數 timer 以及 ticker: 方法。最後還要設定 Slider View 的 IBOutlet 變數。

```
#import <UIKit/UIKit.h>
#import <AVFoundation/AVFoundation.h>

@interface ViewController : UIViewController <AVAudioPlayerDelegate>
{
    AVAudioPlayer *audioPlayer;
    NSTimer *timer;
}
@property (weak, nonatomic) IBOutlet UISlider *slider;

-(void)ticker:(NSTimer *)theTimer;
@end
```

4 開啟 ViewController.m，在 viewDidLoad 方法中，先設定 slider 的最小值、最大值與目前值。其中最大值當然是設定成跟音樂的播放的時間長度一樣，最小值與目前值都設定為 0。當音樂開始播放後，我們啟動計時器（參考先備知識 2），我們設定每隔 1 秒鐘觸發一次 ticker: 即可。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    // 找到 music.mp3 的路徑
    NSString *filePath = [[NSBundle mainBundle] pathForResource:@"music"
                          ofType:@"mp3"];
    NSData *fileData = [NSData dataWithContentsOfFile:filePath];

    // 初始化 audioPlayer
    audioPlayer = [[AVAudioPlayer alloc] initWithData:fileData
                   error:nil];
    // 設定代理人類別
    audioPlayer.delegate = self;

    // 檢查是否一切就緒，如果沒問題就播放音樂
    if (audioPlayer != nil) {
        if ([audioPlayer prepareToPlay]) {
            // 設定 slider 的最小值
            self.slider.minimumValue = 0;
            // 將 slider 的最大值設定成跟音樂的總播放時間一樣
            self.slider.maximumValue = audioPlayer.duration;
            // 設定目前 slider 的值為 0，音樂預設是從頭播放
            self.slider.value = 0;

            [audioPlayer play];

            // 啟動計時器
            timer = [NSTimer scheduledTimerWithTimeInterval:1.0
                                                     target:self
                                                     selector:@selector(ticker:)
                                                     userInfo:nil
                                                     repeats:YES
                     ];
        }
    }
}
```

5 實作 ticker: 方法，這個方法是當計數器被觸發後呼叫的。根據上一步計時器的設定，這個方法每 1 秒鐘會被呼叫一次，因此在這個方法中我們更新 slider 的目前值，讓它跟音樂目前的播放進度一致就好了。

```
-(void)ticker:(NSTimer *)theTimer
{
    self.slider.value = audioPlayer.currentTime;
}
```

6 設定 Slider View 的 IBAction 方法，攔截的事件為 Touch Up Inside，我們希望使用者可以去調整 slider 的目前值，然後跟著改變音樂的播放進度。

```
- (IBAction)changePlayTime:(id)sender
{
    audioPlayer.currentTime = self.slider.value;
}
```

7 執行看看。

7-9 播放已經存在的音樂

難易度：★

大部分的情況下，我們的 iPhone 或 iPad 中已經儲存了許多的音樂，現在要來播放這些音樂。

適用裝置：iPhone / iPad
Framework：MediaPlayer.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 MediaPlayer.framework 加到專案中。

3 開啟 Storyboard，拖放一個按鈕到 View Controller 上，這個按鈕的目的是按下去後可以顯示音樂列表讓使用者選取想要播放的歌曲。

4 開啟 ViewController.h，匯入 MediaPlayer.h 標頭檔。

```
#import <UIKit/UIKit.h>
#import <MediaPlayer/MediaPlayer.h>

@interface ViewController : UIViewController
                                <MPMediaPickerControllerDelegate>

@end
```

5 在 ViewController.m 中建立按鈕的 IBAction 方法，當按鈕按下去後，開啟音樂檔案列表。列表的畫面我們不用花時間去設計，因為 iOS 會呼叫系統內建的列表來顯示。

```
- (IBAction)buttonPress:(id)sender
{
    MPMediaPickerController *mediaPicker = [[MPMediaPickerController
           alloc] initWithMediaTypes:MPMediaTypeAnyAudio];
    mediaPicker.delegate = self;
    // 設定可以多選
    mediaPicker.allowsPickingMultipleItems = YES;
    // 開啓音樂檔案列表視窗
```

```
    [self presentViewController:mediaPicker animated:YES
                                                completion:nil];
}
```

6 在 ViewController.m 中實作 mediaPicker:didPickMediaItems: 方法，這個方法是當使用者從音樂列表上選取了音樂後會呼叫。使用者選取的音樂會放在 mediaItemCollection 參數中傳進來。MPMusicPlayerController 有兩個方法：applicationMusicPlayer 與 iPodMusicPlayer，這兩個方法都可以讓 iPhone 或 iPad 播放音樂，差別是在使用 applicationMusicPlayer 來播放音樂時，相關的控制是在 App 中，如果是 iPodMusicPlayer 則是啟動系統的音樂播放程式，因此音樂一開始播放，就跟我們的 App 無關。可以做個簡單的測試，當我們寫的 App 開始播放音樂後，按下 HOME 鍵將 App 丟到背景去執行，如果是 applicationMusicPlayer 音樂會停止，而 iPodMusicPlayer 則不會。

```
-(void)mediaPicker:(MPMediaPickerController *)mediaPicker
         didPickMediaItems:(MPMediaItemCollection *)mediaItemCollection
{
    // 使用者選擇了音樂後會呼叫此方法
    MPMusicPlayerController *musicPlayer = [MPMusicPlayerController
                                            applicationMusicPlayer];
    [musicPlayer setQueueWithItemCollection:mediaItemCollection];
    [musicPlayer play];

    [self dismissViewControllerAnimated:YES completion:nil];
}
```

7 實作 mediaPickerDidCancel:方法，這個方法是當使用者在音樂列表上按取消鈕後會呼叫的方法，因此我們要在這個方法中關閉音樂列表視窗。

```
-(void)mediaPickerDidCancel:
    (MPMediaPickerController *)mediaPicker
{
    // 使用者取消選取後會呼叫此方法
    [self dismissViewControllerAnimated:
         YES completion:nil];
}
```

8 執行看看。這個 App 需在實體機器上執行。

7-10 取得目前播放中的歌曲資訊

難易度：★★

當播放裝置內已經存在的音樂時，可以取得目前播放中的歌曲資訊。

適用裝置：iPhone / iPad

先備知識：無 Framework：MediaPlayer.framework

步驟與說明

1 建立 Single View Application 專案。

2 將 MediaPlayer.framework 加到專案中。

3 開啟 Storyboard，我們需要的畫面稍微有一點點複雜。先拖放 7 個標籤元件、一個 Image View 元件與一個按鈕。標籤元件中的 4 個只是單純的顯示文字而已，另外 3 個則是之後要用來顯示歌曲名稱、專輯名稱與歌手姓名用的。Image View 用來顯示專輯封面照片。按鈕是用來跳到下一首歌。

4 開啟 ViewController.h，先匯入 MediaPlayer.h 標頭檔。然後宣告一個 MPMusicPlayerController 的實體變數 musicPlayer。接下來就是設定 3 個標籤與 Image View 的 IBOutlet 變數。

```
#import <UIKit/UIKit.h>
#import <MediaPlayer/MediaPlayer.h>

@interface ViewController : UIViewController <MPMediaPickerControllerDelegate>
{
    MPMusicPlayerController *musicPlayer;
}

// 專輯封面
@property (weak, nonatomic) IBOutlet UIImageView *artworkImage;
// 歌曲名稱
@property (weak, nonatomic) IBOutlet UILabel *titleLabel;
// 專輯名稱
@property (weak, nonatomic) IBOutlet UILabel *albumLabel;
// 歌手姓名
@property (weak, nonatomic) IBOutlet UILabel *artistLabel;

@end
```

5 開啟 ViewController.m，為了程式碼簡單起見，我們在 viewDidLoad 方法中使用 MPMediaQuery 類別的 songsQuery 方法一次將裝置內所有的歌曲加到播放清單中。然後利用 KVO（Key-Value Observing）技術設定當播放的歌曲變換時會發送通知到我們指定的方法。最後要記得呼叫 MPMusicPlayerController 類別的 beginGeneratingPlaybackNotifications 方法，這樣才會讓播放的狀態發出訊息。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    musicPlayer = [MPMusicPlayerController applicationMusicPlayer];
    // 將裝置內所有的歌曲放到播放清單中
    [musicPlayer setQueueWithQuery:[MPMediaQuery songsQuery]];

    // 設定訊息通知
    NSNotificationCenter *notice = [NSNotificationCenter defaultCenter];
    // 當播放的歌曲改變時要通知我們（呼叫 nowPlayingItemChanged:方法）
    [notice addObserver:self
                selector:@selector(nowPlayingItemChanged:)
                    name:MPMusicPlayerControllerNowPlayingItemDidChangeNotification
                  object:musicPlayer
```

```
        ];

    // 要求歌曲播放時發送一些狀態訊息
    [musicPlayer beginGeneratingPlaybackNotifications];

    // 播放音樂
    [musicPlayer play];
}
```

6 在 ViewController.m 中實作 nowPlayingItemChanged: 方法，這個方法就是上一步利用 KVO 技術註冊的方法。在這個方法中，取得目前播放清單中的歌曲資訊，這個資訊會被封裝在 MPMediaItem 類別中。我們很簡單的只取出其中四個資訊：專輯封面圖片、歌曲名稱、專輯名稱與歌手姓名。

```
-(void) nowPlayingItemChanged:(id)notification
{
    // 當播放的歌曲改變時會呼叫此方法
    MPMediaItem *item = [musicPlayer nowPlayingItem];

    // 取得專輯封面圖片
    MPMediaItemArtwork *artwork = [item valueForProperty:
                                        MPMediaItemPropertyArtwork];
    if (artwork)
        self.artworkImage.image = [artwork imageWithSize:CGSizeMake(130, 130)];

    // 取得歌曲名稱
    self.titleLabel.text = [item valueForProperty:MPMediaItemPropertyTitle];
    // 取得專輯名稱
    self.albumLabel.text = [item valueForProperty:MPMediaItemPropertyAlbumTitle];
    // 取得歌手姓名
    self.artistLabel.text = [item valueForProperty:MPMediaItemPropertyArtist];
}
```

7 在 ViewController.m 中設定按鈕的 IBAction 方法，在這個方法中呼叫 skipToNextItem 來跳到下一首歌曲。同樣的，可以呼叫 skipToPreviousItem 回到上一首歌，呼叫 skipToBeginning 讓現在播放的歌曲再從頭開始播。

```
- (IBAction)nextSong:(id)sender
{
    // 下一首歌曲
    [musicPlayer skipToNextItem];
    [musicPlayer play];
}
```

8 執行看看。