

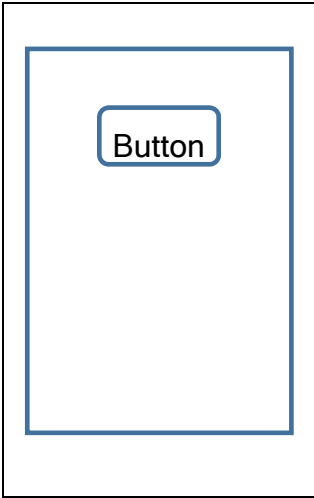
CHAPTER 6-7

AudioPlayer：音效播放

播放 mp3 音效

學習概念：

1. 首先用 IB 建立【操作按鈕】。
2. 將【操作按鈕】「Button」與【程式碼】連結。
3. 最後在實作檔中相關程式，於處理載入後所觸發的事件，也就是撰寫利用【操作按鈕】結合【程式碼】及撰寫程式，讓【操作按鈕】播放音樂。

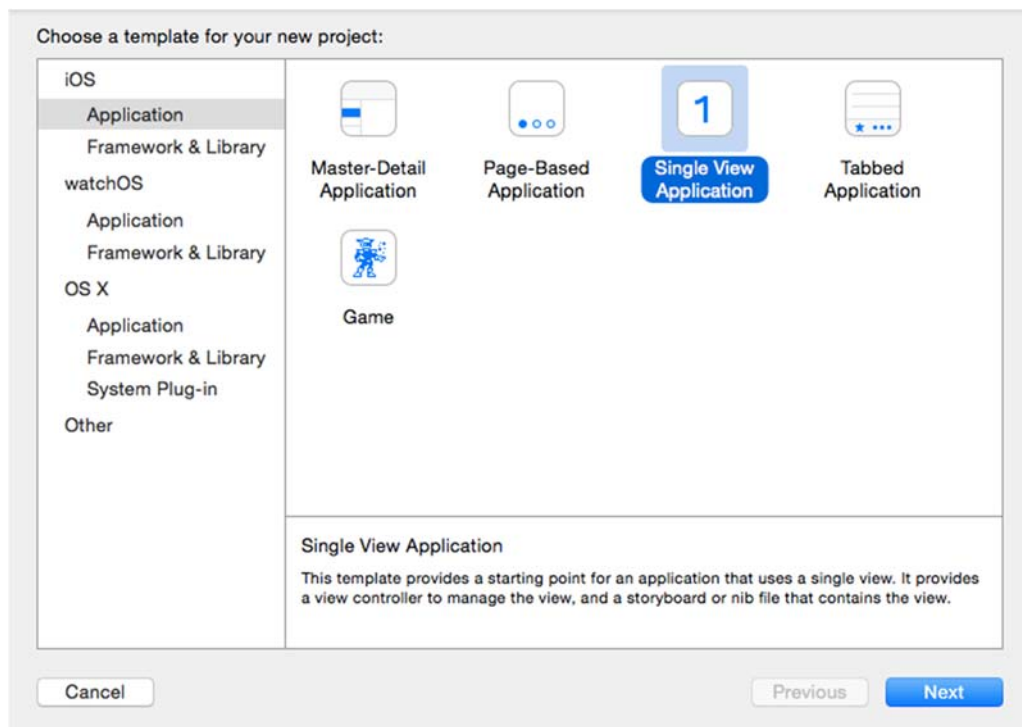
Interface Builder	原始碼編輯器
<p>▼設計畫面</p> 	<pre>@IBAction func tapBtn(sender: AnyObject){ }</pre> <p>▼設定文字標籤顯示現在的時間</p> <pre>let audioPath = NSBundle.mainBundle().pathForResource("alarm", ofType: "mp3") let alarmSound = NSURL(fileURLWithPath: audioPath!) do{ audioPlayer = try AVAudioPlayer(contentsOfURL:alarmSound) audioPlayer.prepareToPlay() audioPlayer.play() }catch { print("Error getting the audio file") }</pre>

【執行結果】

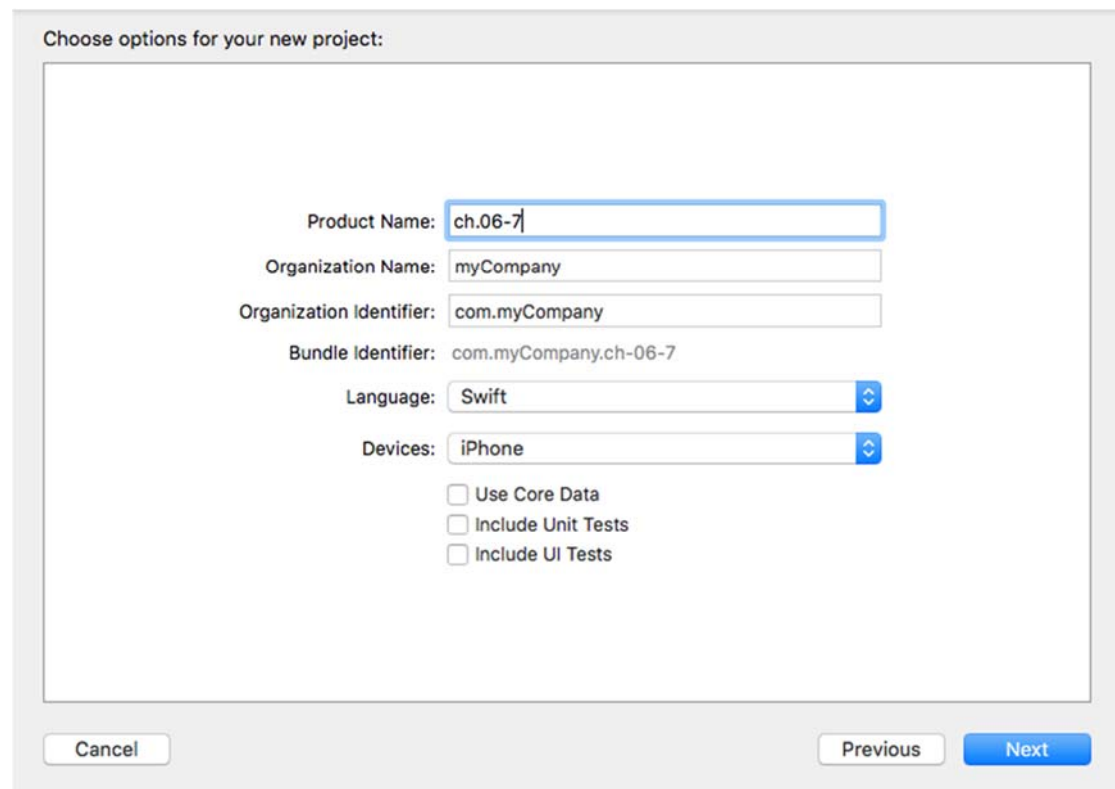
當 App 執行後，將自動播放音效。

Step.1

開啟 **xcode** 時會出現的畫面，點選 **iOS** 下的【**Application**】，接著右視窗選擇【**Single View Application**】，點選【**Next**】選項後進入設定的基本視窗。



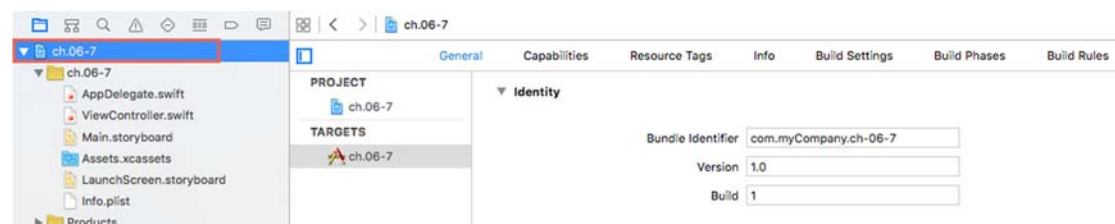
檔名及名稱設定，請將【Product Name】設定為 ch.06-7



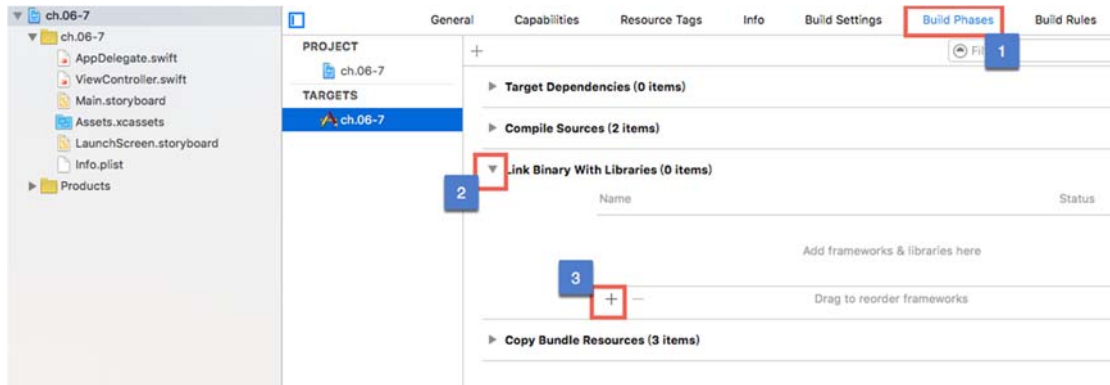
Step.2

加入音效的函示庫於專案中

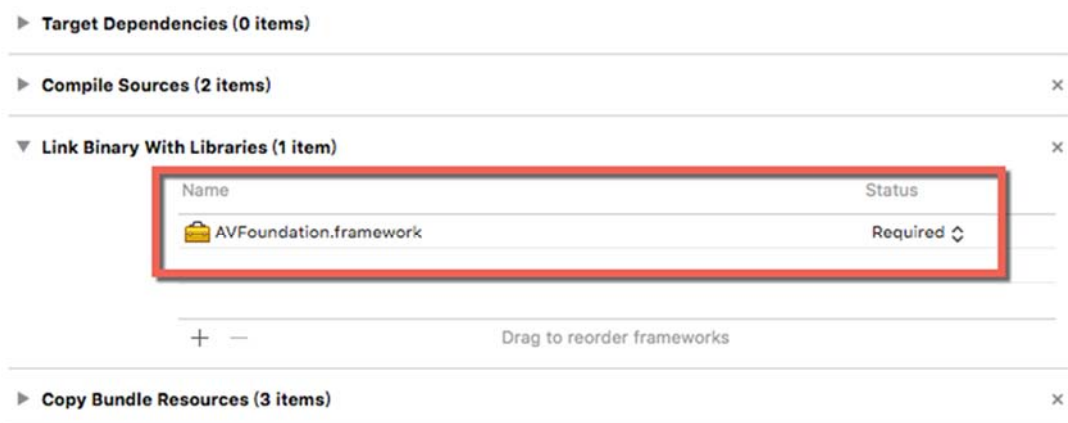
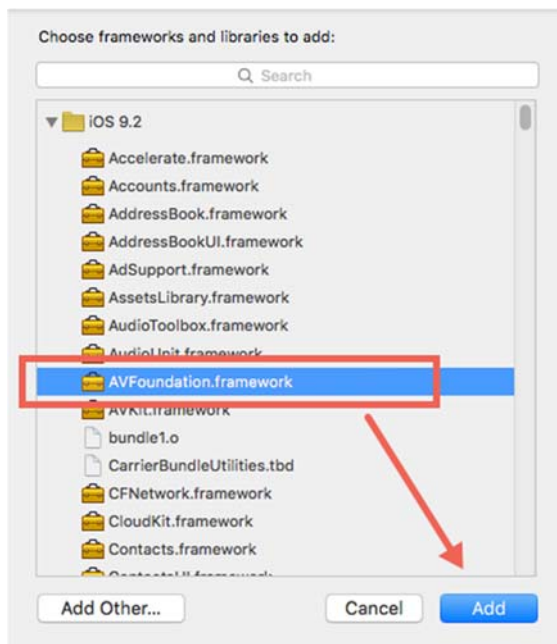
1. 點選左方 專案的最上層，右方會顯示整體專案的相關資訊。



2. 切換至 **Build Phases** 頁籤，並展開 **Link Binary With Libraries**，選取下方的『+』進行新增函示庫。

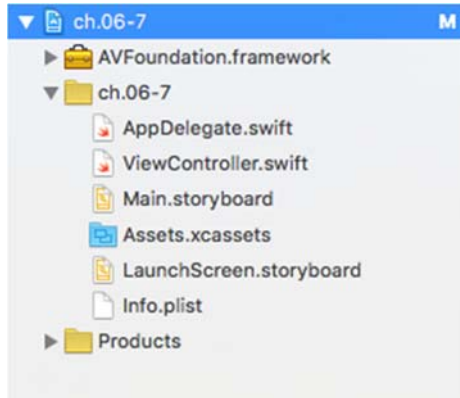


3. 由所列出的函式中選擇 AVFoundation.framework，選擇 Add 加入。



Step.3

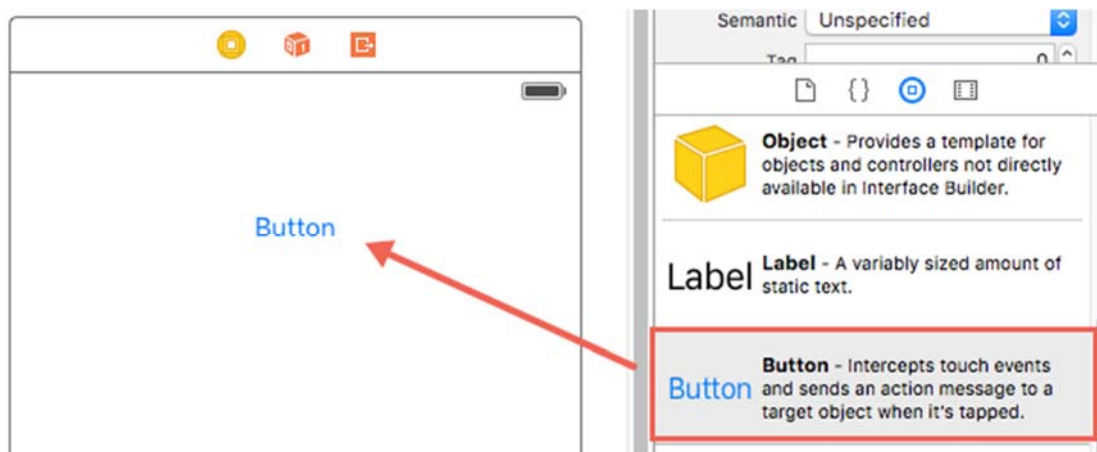
選取【Main.storyboard】



本章操作已選擇【iPhone 4.7-inch】 操作頁面。(詳見 5-1 屬性設定小技巧)

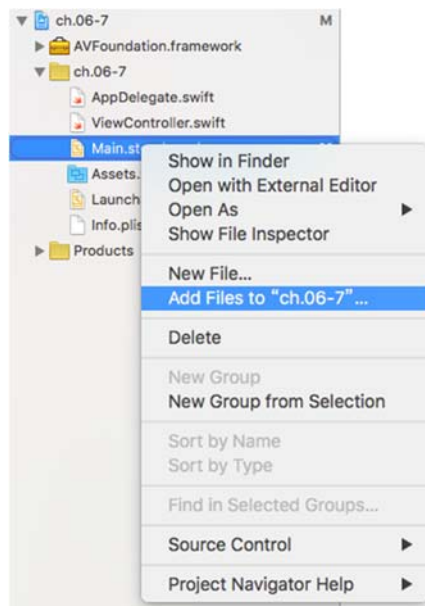
Step.4

從【物件區】中拖曳【操作按鈕】到 IB 畫面中。

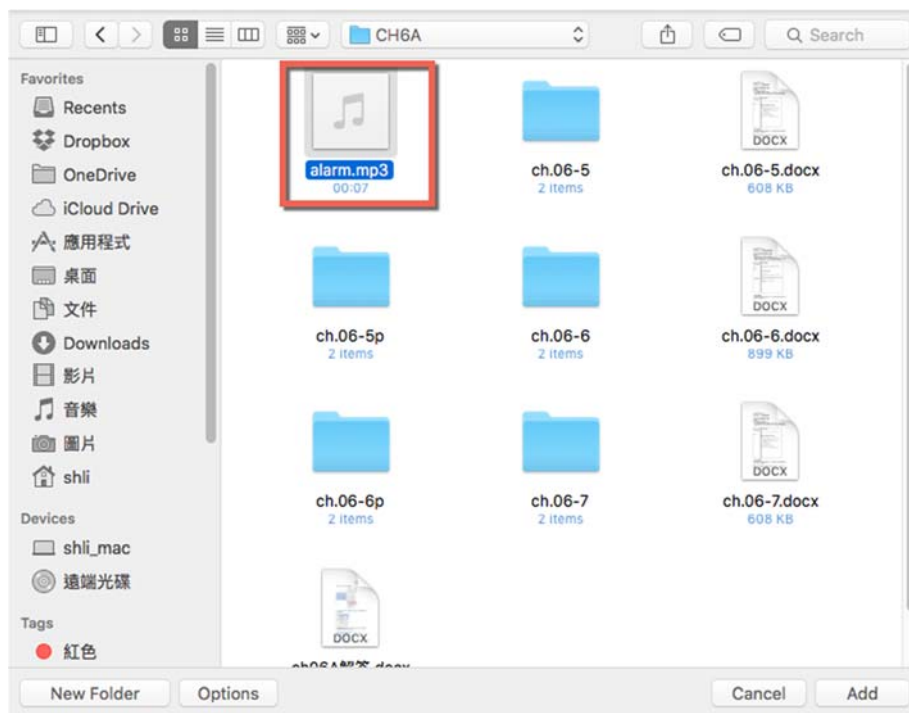


Step.5

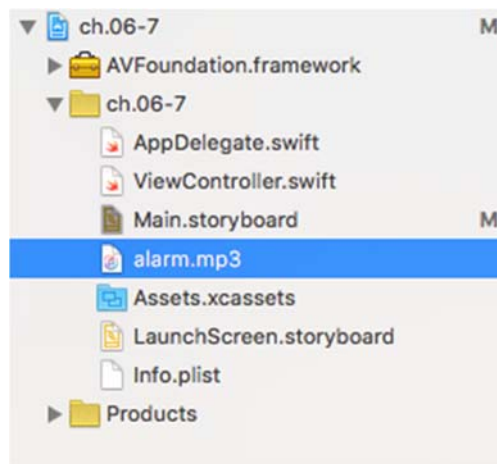
- (1) 在這裡我們選擇一個音效檔，展開 ch.06-7 目錄並且在此目錄中任一位置按右鍵→Add Files to “ch.06-7....”。



(2) 在這裡我們選擇選取的檔案，名稱為<alarm.mp3>，並在選取後按「Add」。



(3) 完成後，接著會看到此 alarm.mp3，顯示在 ch.06-7 目錄底下。



(4) 或者，讀者們可以利用拖曳的方式，將音效檔加入到專案內。

將要加入的音效檔案，選取後用拖曳方式加入

Step.6.

接著點選右上方工具列視窗【輔助編碼器】，就是雙圈符號【2】，進程式碼編輯。

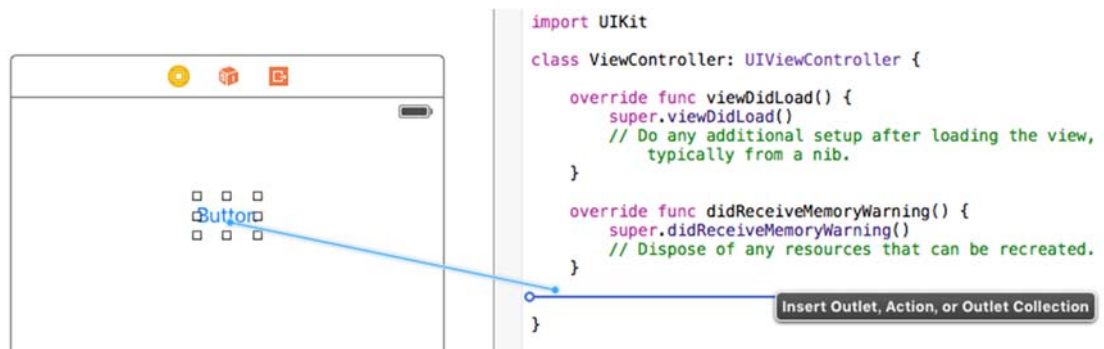


(詳見 5-1 屬性設定小技巧)

Step.7

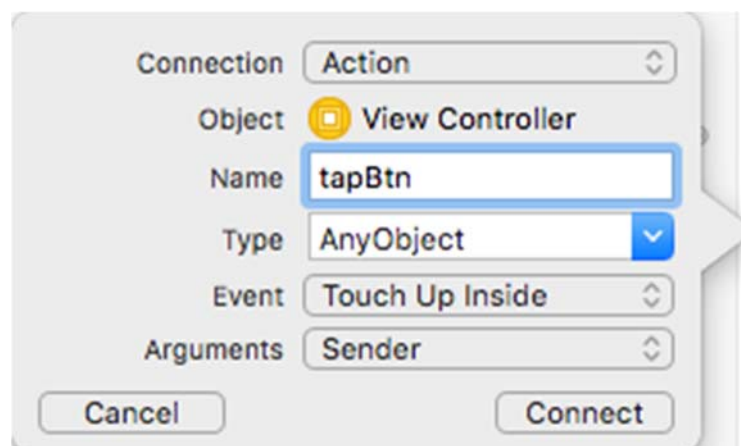
(1) 透過連結即產生的程式碼，控制 IB 建立的元件和【程式碼】連結。

(2) 將【操作按鈕】「Button」與「變數名稱」連結。



按住【Control】用滑鼠拖曳 Button 元件。

在「Connection」欄位點選【Action】、「Name」欄位設定名稱【tapBtn】，按【Connect】按鈕。



在拖曳後自動彈跳的視窗

(3) 將會自動插入程式碼，作為與 IB 的連結。


```

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view,
        typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    @IBAction func tapBtn(sender: AnyObject) {
    }

}

```

連結後出現的程式碼。

Step.8

(1) 接下來，我們在【紅框】中加入設定 **swift** 的程式碼。

在程式最上方需要將 **AVFoundation** 函式庫加上。

```
import AVFoundation
```

接下來宣告一個 **AVAudioPlay** 的物件

```
var audioPlayer = AVAudioPlayer()
```

在 **tapBtn**【操作按鈕】的程式中則需要指定音效檔的名稱與路徑，並由此路徑中取得音效檔。

```
let audioPath = NSBundle.mainBundle().pathForResource("alarm",
ofType: "mp3")
```

```
let alarmSound = NSURL(fileURLWithPath: audioPath!)
```

緊接著進行音效播放的作業，但因為擔心無法有效播放音效檔，因此

Swift2 為了錯誤處理，設立了一種新的機制。要透過

do{ try... } catch{ } 的語法

嘗試執行音效播放，若失敗則告知無法取得音效檔

```
do{
    audioPlayer = try AVAudioPlayer(contentsOfURL:alarmSound)
    audioPlayer.prepareToPlay()
    audioPlayer.play()
}catch {
    print("Error getting the audio file")
}
```

```
import UIKit
import AVFoundation

class ViewController: UIViewController {
    var audioPlayer = AVAudioPlayer()

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    @IBAction func tapBtn(sender: AnyObject) {
        let audioPath = NSBundle.mainBundle().pathForResource("alarm", ofType: "mp3")
        let alarmSound = NSURL(fileURLWithPath: audioPath!)

        do{
            audioPlayer = try AVAudioPlayer(contentsOfURL:alarmSound)
            audioPlayer.prepareToPlay()
            audioPlayer.play()
        }catch {
            print("Error getting the audio file")
        }
    }
}
```

Step.9

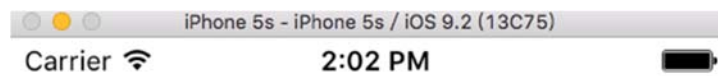
在上方工具列按下【執行鍵▶】（Build and then run the current scheme），啟動模擬器執行程式。



Step.10

當 App 啟動後在顯示畫面時，畫面內容在出現【操作按鈕】「Button」，按下後

就會播放音效。



Button



自我練習

接下呢，我們要練習的是設計一個鬧鐘，使用 `DatePicker` 調撥時間，按下 `set Alarm` 設定鬧鐘時間，當時間到時就響鈴，按下 `Alarm OFF` 則停止鬧鈴。

。