

Class 2



Interface Control

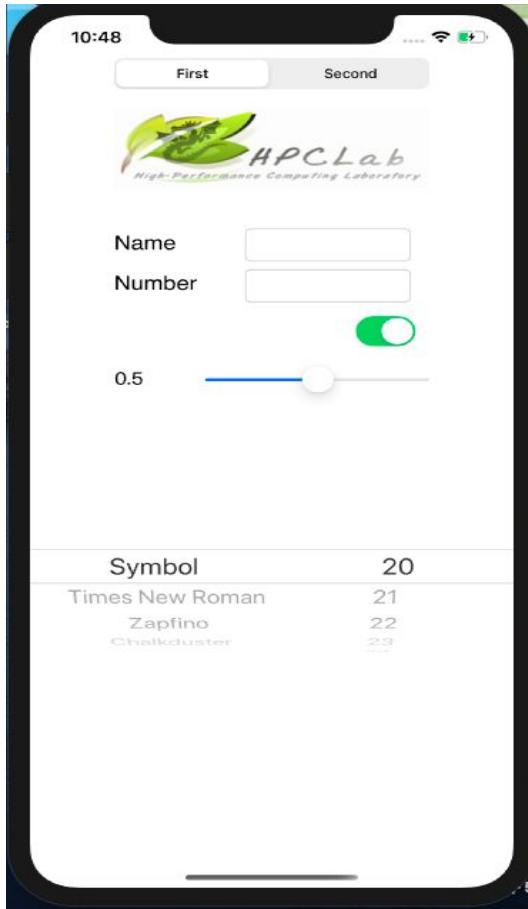


Image View

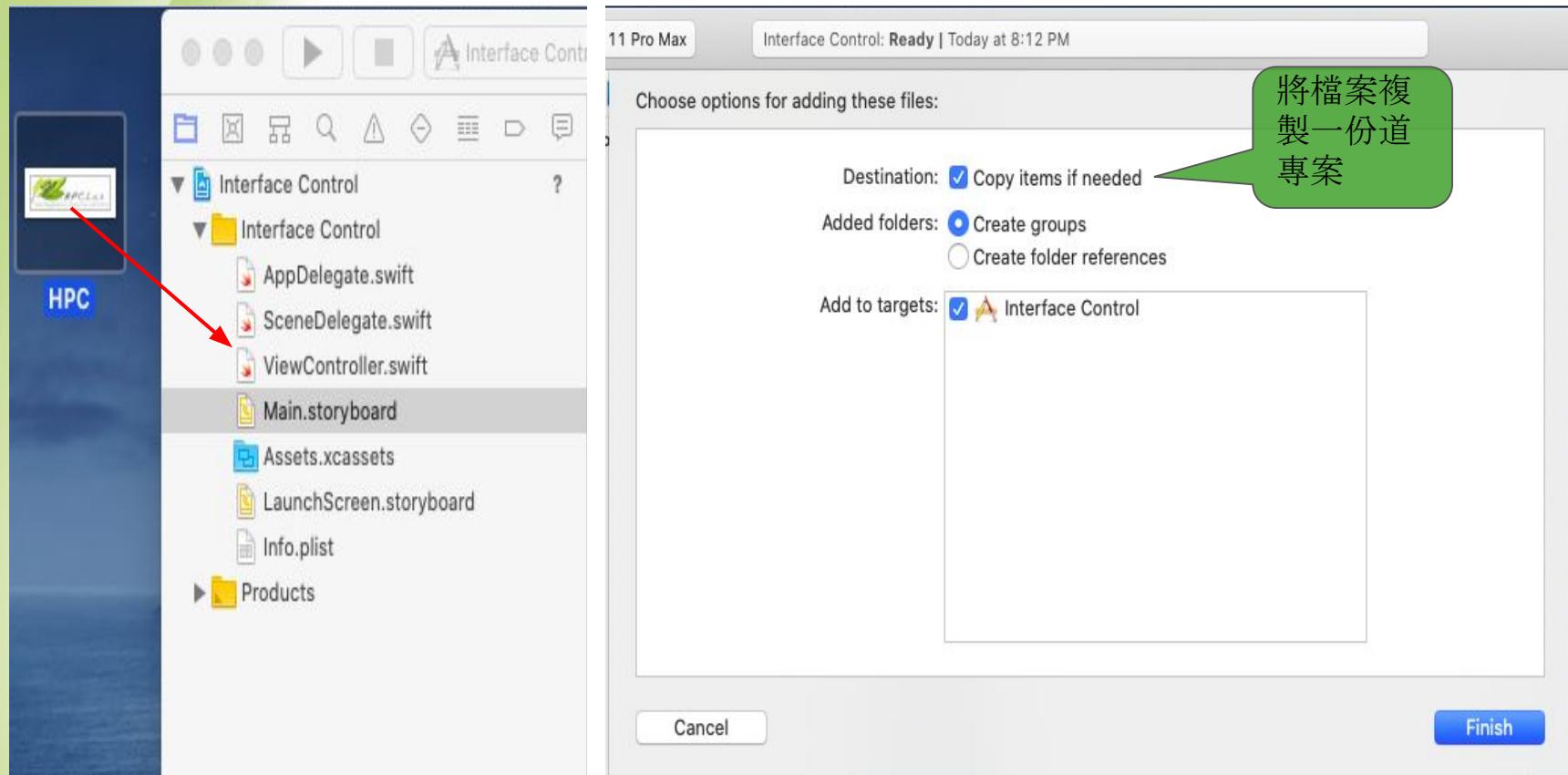


Image View

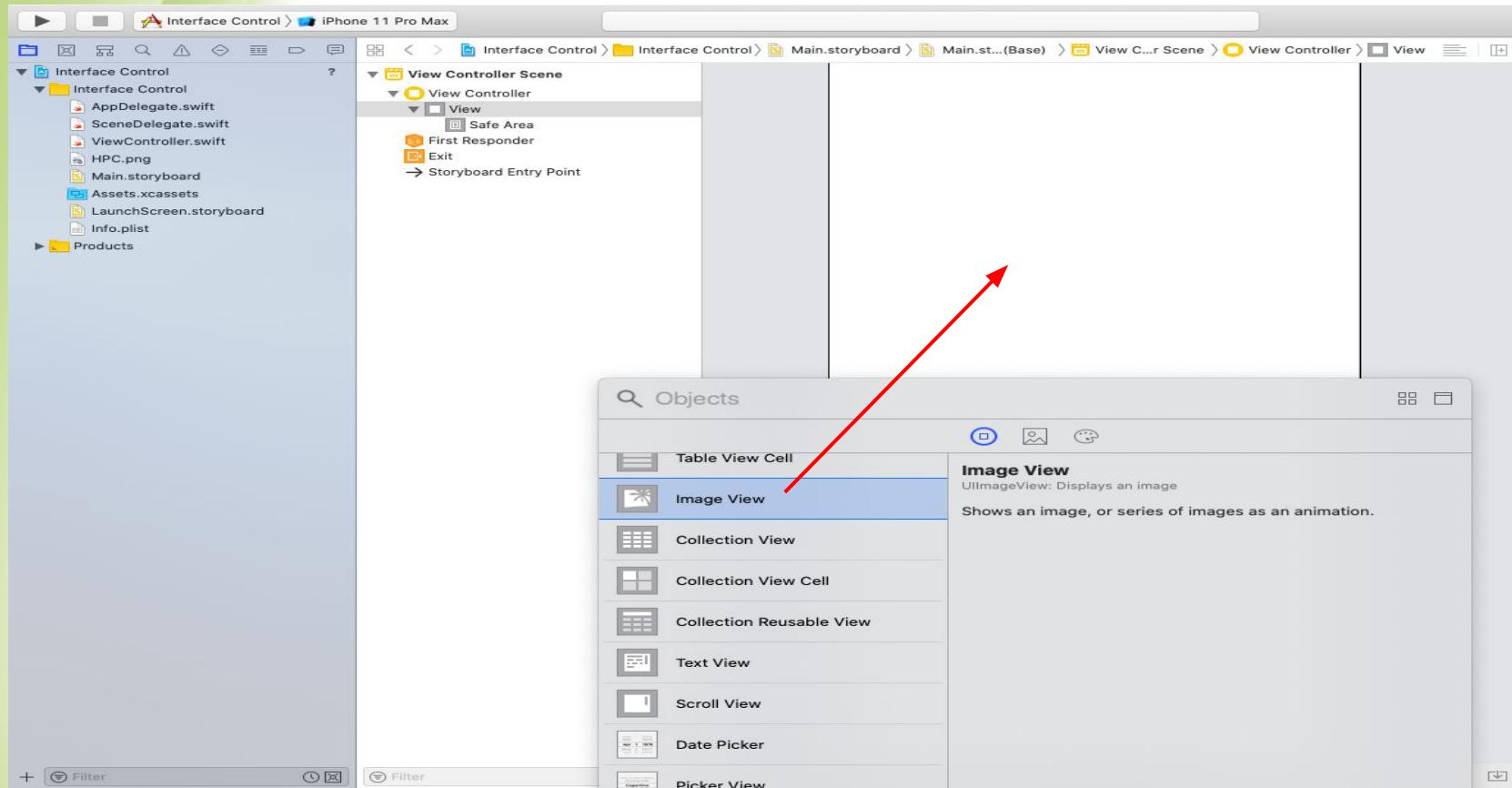


Image View

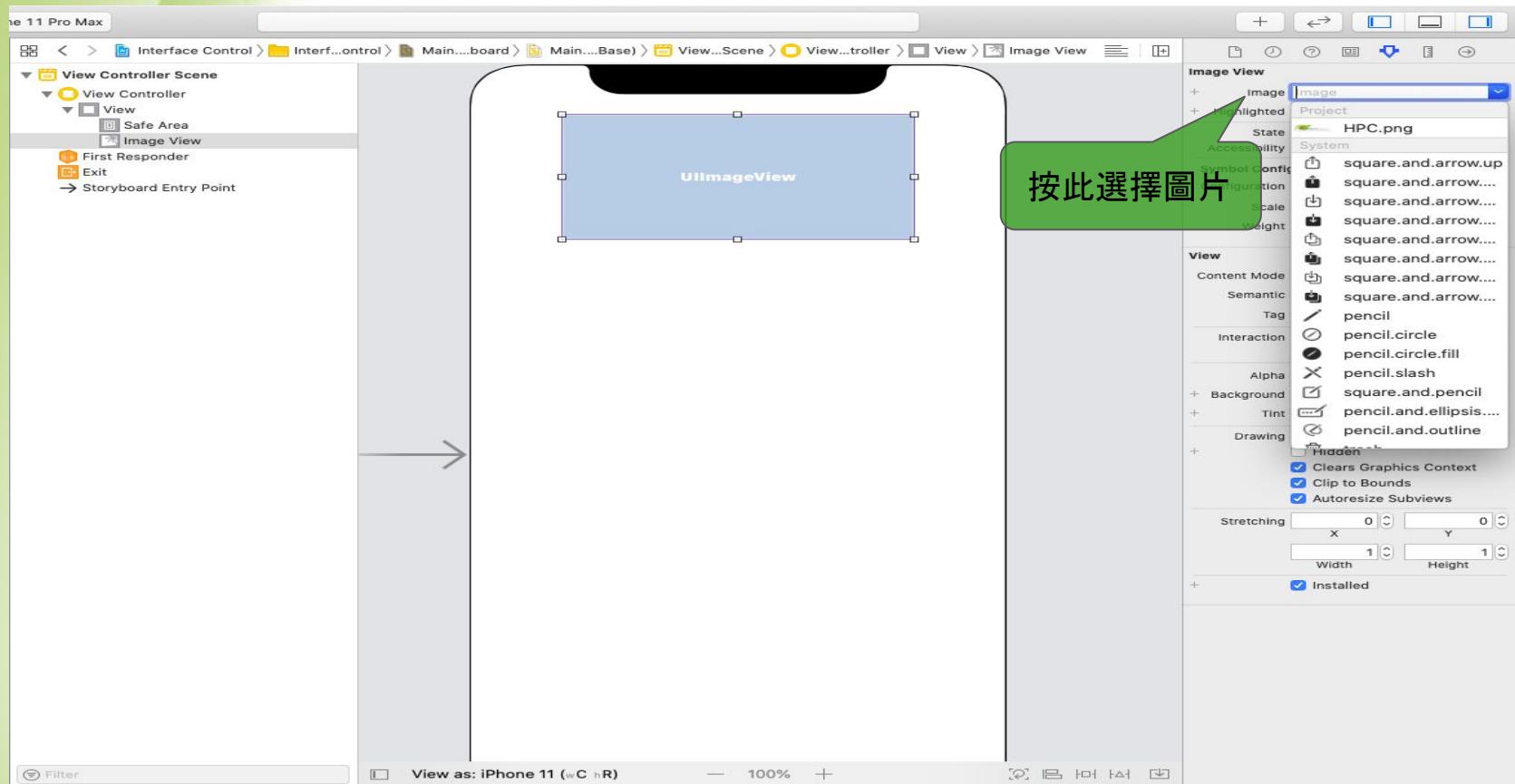


Image View

The screenshot shows the Xcode interface with three main panes:

- Left Pane (Main.storyboard):** Shows the storyboard file structure. A context menu is open over a view controller, with the "Connections Inspector" tab selected. The "Object" dropdown is set to "View Controller". The "Name" field is empty, indicated by a red arrow. The "Type" dropdown is set to "UIImageView" and the "Storage" dropdown is set to "Weak".
- Middle Pane (ViewController.swift):** Displays the Swift code for the ViewController. A red arrow points from the "Name" field in the Connections Inspector to the "UIViewController" line in the code.
- Right Pane (iPhone 11 Pro Max Simulator):** Shows the simulator displaying the "HPCLab" logo with a green leaf and the text "High-Performance Computing Laboratory".

```
1 //  
2 //  ViewController.swift  
3 //  Interface Control  
4 //  
5 //  Created by evan on 2020/3/3.  
6 //  Copyright © 2020 evan. All  
7 //  rights reserved.  
8 //  
9 import UIKit  
10  
class ViewController:  
    UIViewController {  
        override func viewDidLoad() {  
            super.viewDidLoad()  
            // Do any additional  
            // setup after loading  
            // the view.  
        }  
        16  
        17  
        18  
        19    }  
        20  
        21
```

Image View

```
8  
9 import UIKit  
10  
11 class ViewController: UIViewController {  
12  
13     @IBOutlet weak var HPC: UIImageView!  
14  
15     override func viewDidLoad() {  
16         super.viewDidLoad()  
17         // Do any additional setup after loading the view.  
18         HPC.contentMode = .sc  
19             ⓘ Type 'UIView.ContentMode' has no member 'sc'  
UIView.ContentMode scaleAspectFill")  
UIView.ContentMode scaleToFill  
UIView.ContentMode scaleAspectFit
```

scale aspect fill:
保持長寬比的
原則下，縮放到
充滿整個 view

scale aspect fill:
保持長寬比的
原則下，縮放到
充滿整個 view

Scale To Fill:縮放
圖片讓圖片充滿
view

The option to scale the content to fit the size of
the view by maintaining the aspect ratio. Any
remaining area of the view's bounds is ...

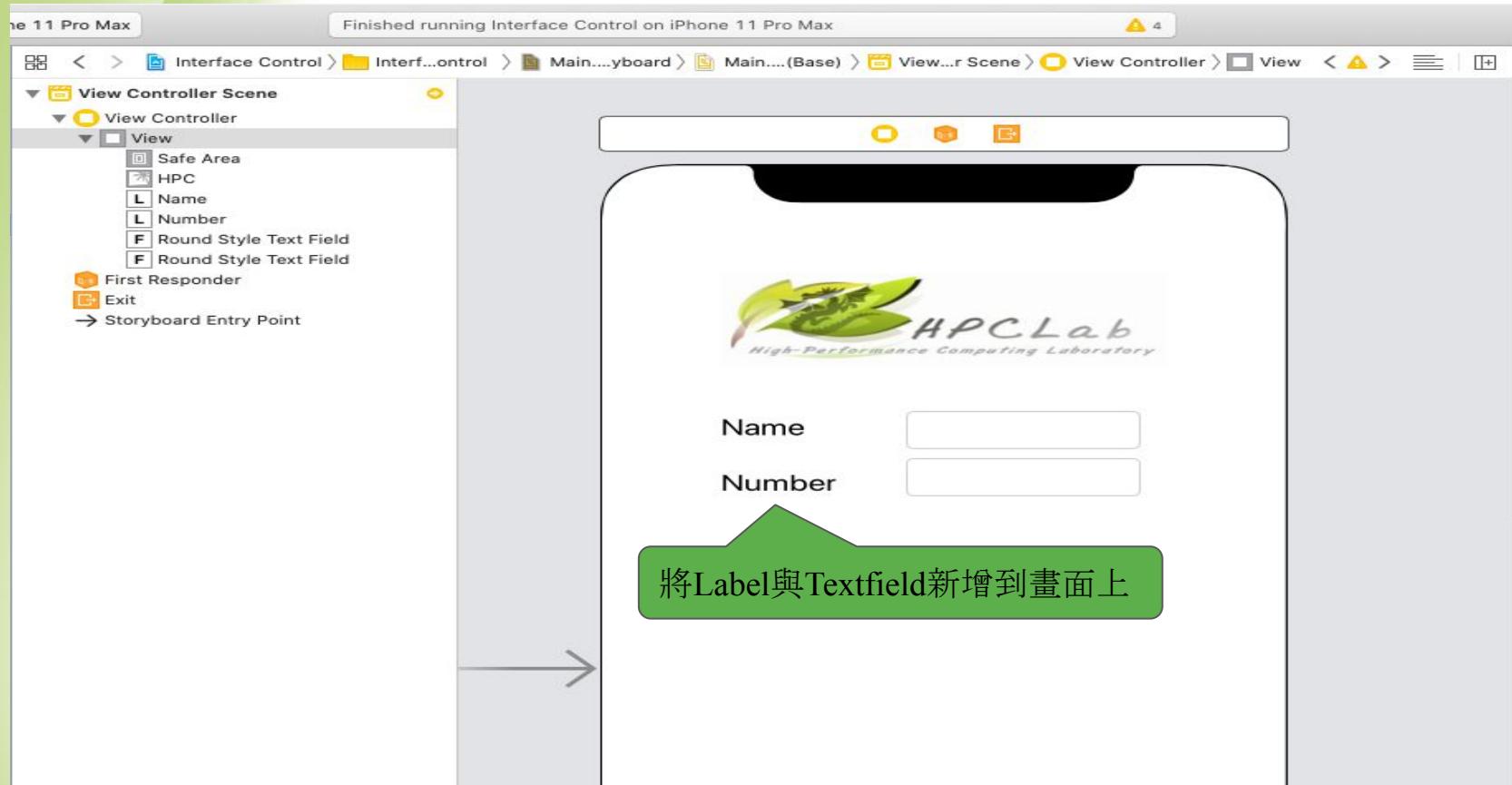
Image View

```
iPhone 11 Pro Max Finished running Interface Control on iPhone 11 Pro Max ▲ 1 ⚡ 1
File Interface Control Interface Control ViewController.swift viewDidLoad()
1 // ViewController.swift
2 // Interface Control
3 // Created by evan on 2020/3/3.
4 // Copyright © 2020 evan. All rights reserved.
5 //
6
7 //
8
9 import UIKit
10
11 class ViewController: UIViewController {
12     @IBOutlet weak var HPC: UIImageView
13
14     override func viewDidLoad() {
15         super.viewDidLoad()
16         // Do any additional setup after loading the view.
17         HPC.contentMode = .scaleAspectFit
18         HPC.image = UIImage(named: "HPC.png")
19     }
20 }
21
22
23 }
24
25 }
```

在App畫面出現前的初始化

設置玩顯示模式必須要重載圖片來達到設定

KeyBoard Control



KeyBoard Control

The screenshot shows the Xcode Interface Builder environment with a storyboard scene for an iPhone 11 Pro Max. The scene contains a view controller with a logo and two text input fields labeled "Name" and "Number". A green callout bubble points to the "Number" text field, containing the Chinese text "將鍵盤設置成數字鍵盤". The right side of the interface shows the "Text Field" settings panel, which includes options for Text (Plain), Color (Default (Label Color)), Font (System 14.0), and various traits like Content Type (Unspecified) and Keyboard Type (Number Pad). The "Adjust to Fit" checkbox is checked under the font section.

iPhone 11 Pro Max

Finished running Interface Control on iPhone 11 Pro Max

View Controller Scene

- View Controller
- View
- Safe Area
- HPC
- Name
- Number
- Round Style Text Field
- Round Style Text Field

First Responder

Exit

Storyboard Entry Point

Name

Number

將鍵盤設置成數字鍵盤

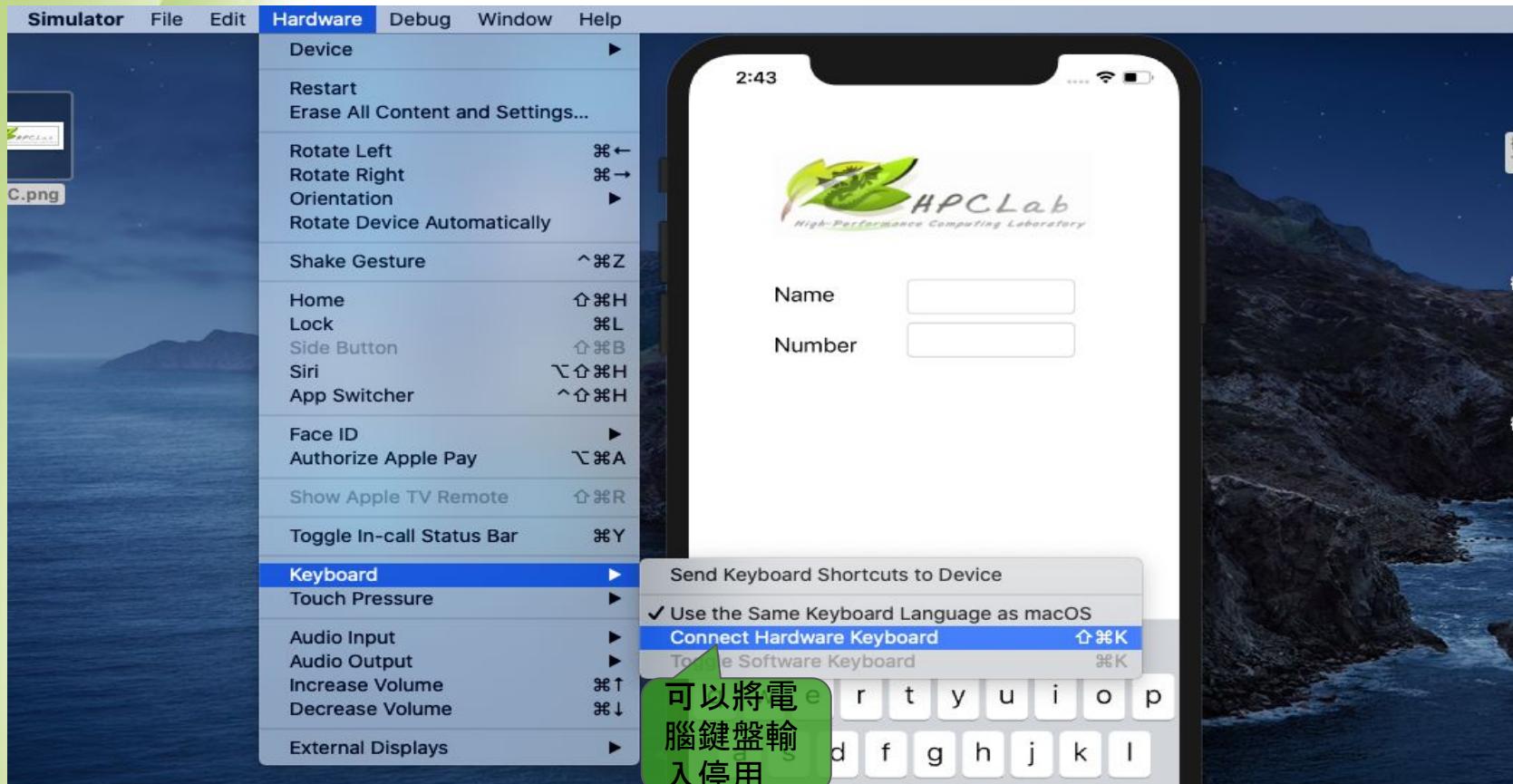
Text Field

- Text: Plain
- Color: Default (Label Color)
- Font: System 14.0
- Dynamic Type: Automatically Adjusts Font
- Alignment: Placeholder Text
- Background:
- Disabled:
- Border Style:
- Clear Button: Never appears
- Min Font Size: 17
- Adjust to Fit
- Text Input Traits
- Content Type: Unspecified
- Capitalization: None
- Correction: Default
- Smart Dashes: Default
- Smart Insert: Default
- Smart Quotes: Default
- Spell Checking: Default
- Keyboard Type: Number Pad
- Keyboard Look: Default
- Return Key: Default
- Auto-enable Return Key
- Secure Text Entry

KeyBoard Control



KeyBoard Control



可以將電腦
鍵盤輸入停用

KeyBoard Control

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows files like Interface Control, iPhone 11 Pro Max, AppDelegate.swift, SceneDelegate.swift, ViewController.swift, Main.storyboard, Assets.xcassets, LaunchScreen.storyboard, Info.plist, and Products.
- Code Editor:** Displays Swift code for a ViewController:

```
6 // Copyright © 2020 evan. All rights reserved.  
7 //  
8  
9 import UIKit  
10  
11 class ViewController: UIViewController {  
12  
    @IBOutlet weak var HPC: UIImageView!  
    @IBOutlet weak var nameTextField:  
        UITextField!  
    @IBOutlet weak var numTextField:  
        UITextField!  
15  
20  
25  
26  
27  
28  
29  
30  
31
```
- Assistant Editor:** Shows the storyboard preview for an iPhone 11 Pro Max. It features an HPC logo at the top, followed by two text fields labeled "Name" and "Number". A red arrow points from the storyboard to the "nameTextField" outlet in the code editor.
- Object Library:** A modal window titled "Connect" is open, showing the connection settings for the "nameTextField" outlet. The "Object" dropdown is set to "View Controller". The "Name" dropdown is set to "nameTextField". The "Type" dropdown is set to "UITextField". The "Event" dropdown is set to "Did End On Exit". A green callout bubble with the text "將事件選為 Did End On Exit (按下return觸發的事件)" is overlaid on the bottom left of the modal.

KeyBoard Control

```
8  
9 import UIKit  
10  
11 class ViewController: UIViewController {  
12  
    @IBOutlet weak var HPC: UIImageView!  
    @IBOutlet weak var nameTextField: UITextField!  
    @IBOutlet weak var numTextField: UITextField!  
16  
    @IBAction func nameTextField(_ sender: UITextField) {  
18        sender.resignFirstResponder()  
19    }  
20    將TextField取消  
    FirstResponder  
    因此鍵盤會消失  
21}
```

當我們按下nameTextField時他變為FirstResponder因此開啟鍵盤

但是數字鍵盤
沒有return因此要使用另一種方法將它縮小

KeyBoard Control

```
iPhone 11 Pro Max Finished running Interface Control on iPhone 11 Pro Max
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
```

```
UIViewController {
    @IBOutlet weak var HPC: UIImageView!
    @IBOutlet weak var nameTextField: UITextField!
    @IBOutlet weak var numTextField: UITextField!

    @IBAction func nameTextField(_ sender: UITextField) {
        sender.resignFirstResponder()
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
        HPC.contentMode = .scaleAspectFit
        HPC.image =
    }
}
```

The storyboard shows a single view controller with a logo at the top and two text fields below it, labeled "Name" and "Number". A red arrow points from the line 20 in the code to the "Name" text field in the storyboard.

Outlet Collections
gestureRecognizers

Sent Events

- Did End On Exit
- Editing Changed
- Editing Did Begin
- Editing Did End
- Primary Action Triggered
- Touch Cancel
- Touch Down
- Touch Down Repeat
- Touch Drag Enter
- Touch Drag Exit
- Touch Drag Inside
- Touch Drag Outside
- Touch Up Inside
- Touch Up Outside
- Value Changed

Referencing Outlets

2. 將 Touch Down Event 連結到 View Controller

1. 先選取 View 並
顯則 connections
設定

KeyBoard Control

```
iPhone 11 Pro Max Running Interface Control on iPhone 11 Pro Max □ 4  
Interface Control > Interface Control > ViewController.swift > ViewController  
8  
9 import UIKit  
10  
11 class ViewController: UIViewController {  
12  
13     @IBOutlet weak var HPC: UIImageView!  
14     @IBOutlet weak var nameTextField: UITextField!  
15     @IBOutlet weak var numTextField: UITextField!  
16  
17     @IBAction func nameTextField(_ sender: UITextField) {  
18         sender.resignFirstResponder()  
19     }  
20  
21     當背景被按下時將  
22     numTextField取消  
23     FirstResponder  
24  
25     @IBAction func bgTouchDown(_ sender: UIControl) {  
26         numTextField.resignFirstResponder()  
27     }
```

當背景被按下時將
numTextField取消
FirstResponder

Using Slider



Using Slider

```
 15 @IBOutlet weak var numTextField:  
     UITextField!  
 16  
 17 @IBOutlet weak var alpha:  
     UILabel!  
 18  
 19 @IBAction func nameTextField(_  
     sender: UITextField) {  
     sender.resignFirstResponder()  
}  
 20  
 21  
 22 @IBAction func bgTouchDown(_  
     sender: UIControl) {  
 23  
     numTextField  
     .resignFirstResponder()  
}  
 24  
 25  
 26 @IBAction func alphaSlider(_  
     sender: UISlider) {  
 27  
}  
 28  
 29  
 30  
 31
```



Using Slider

Interface Control on iPhone 11 Pro Max

Mai...ard > Mai...ase) > Vie...ene > Vie...oller > Control > Horizontal Slider < ▲ ▼ > □ □ □ □

The screenshot shows a storyboard for an iPhone 11 Pro Max application. The main view contains three text fields labeled "Name", "Number", and "Label", and a horizontal slider. A green callout bubble points to the "Slider" section of the right-hand Inspector panel, specifically highlighting the "Value", "Minimum", and "Maximum" fields. These fields are currently set to 0.5, 0, and 1 respectively. The Inspector panel also displays settings for "Min Image", "Max Image", "Min Track", "Max Track", and "Thumb Tint". The "Events" section has "Continuous Updates" checked. The "Control" section shows alignment settings for both horizontal and vertical orientations. Under "State", "Enabled" is checked. The "View" section includes "Content Mode" set to "Scale To Fill" and "Semantic" set to "Unspecified".

此處可以修改最大最小值

Slider

Value: 0.5

Minimum: 0

Maximum: 1

Min Image:

Max Image:

Min Track: Default

Max Track: Default

Thumb Tint: Default

Events: Continuous Updates

Control

Alignment: Horizontal
Vertical

State: Selected
 Enabled
 Highlighted

View

Content Mode: Scale To Fill

Semantic: Unspecified

Using Slider

```
class ViewController: UIViewController {  
    12  
    @IBOutlet weak var HPC: UIImageView!  
    @IBOutlet weak var nameTextField: UITextField!  
    @IBOutlet weak var numTextField: UITextField!  
    16  
    @IBOutlet weak var alpha: UILabel!  
    18  
    @IBAction func nameTextField(_ sender: UITextField) {  
        sender.resignFirstResponder()  
    }  
    21  
    22  
    @IBAction func bgTouchDown(_ sender: UIControl) {  
        numTextField.resignFirstResponder()  
    }  
    25  
    26  
    @IBAction func alphaSlider(_ sender: UISlider) {  
        alpha.text = String(sender.value)  
    }  
    28  
    29  
    30
```

當滑動Slider
時將Label的
值改變

Using Slider



Using Slider

Interface Control | Build Interface Control: Failed | Today at 7:29 PM

5 1

```
class ViewController: UIViewController {
    @IBOutlet weak var HPC: UIImageView!
    @IBOutlet weak var nameTextField: UITextField!
    @IBOutlet weak var numTextField: UITextField!
    @IBOutlet weak var alpha: UILabel!
    @IBAction func nameTextField(_ sender: UITextField) {
        sender.resignFirstResponder()
    }
}
```

需要得到拉霸的起始值因此要將拉霸作為Outlet連結過去

需要得到拉霸的起始值因此要將拉霸作為Outlet連結過去



```
@IBAction func  
    nameTextField(_ sender:  
    UITextField) {  
  
    sender  
.resignFirstResponder  
    ()
```

Using Slider

```
@IBAction func alphaSlider(_ sender: UISlider) {  
    alpha.text = String(format: "%.1f", sender.value)  
    HPC.alpha = CGFloat(sender.value)  
    HPC.image = UIImage(named: "HPC.png")
```

圖片的alpha
值為CGFloat
型態因此要
將型態轉為
CGFloat
}

將Label顯示
的數字設為
小數後一位

```
35     override func viewDidLoad() {  
36         super.viewDidLoad()  
37         // Do any additional setup after loading the view.  
38         HPC.contentMode = .scaleAspectFit  
39         HPC.image = UIImage(named: "HPC.png")  
40         alpha.text = String(format: "%.1f", sliderValue.value)|  
41     }  
42 }
```

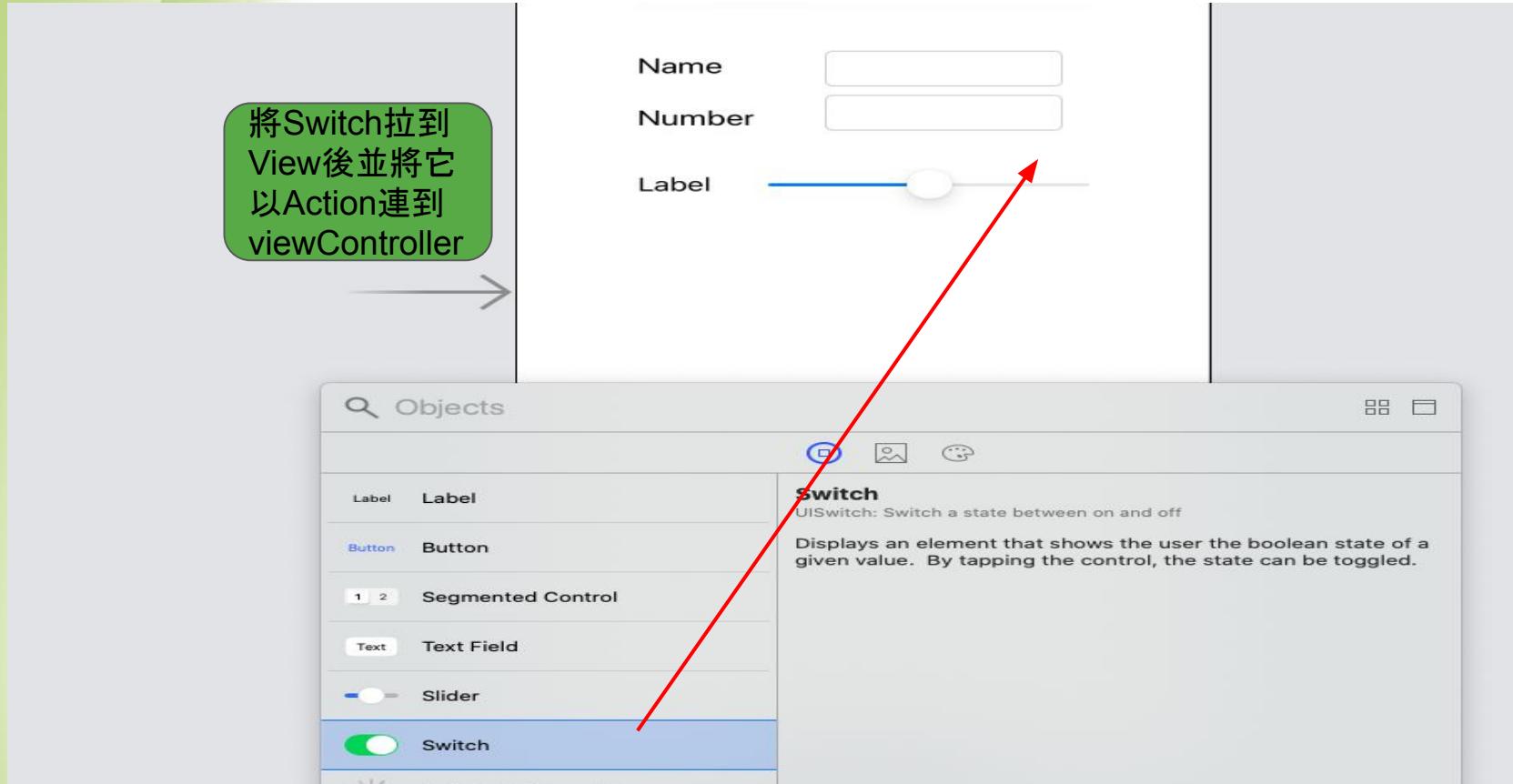
設定alpha
值後要再重
設圖片來刷
新

在viewDidLoad中
設定Label的初值

Switch Button



Switch Button



Switch Button

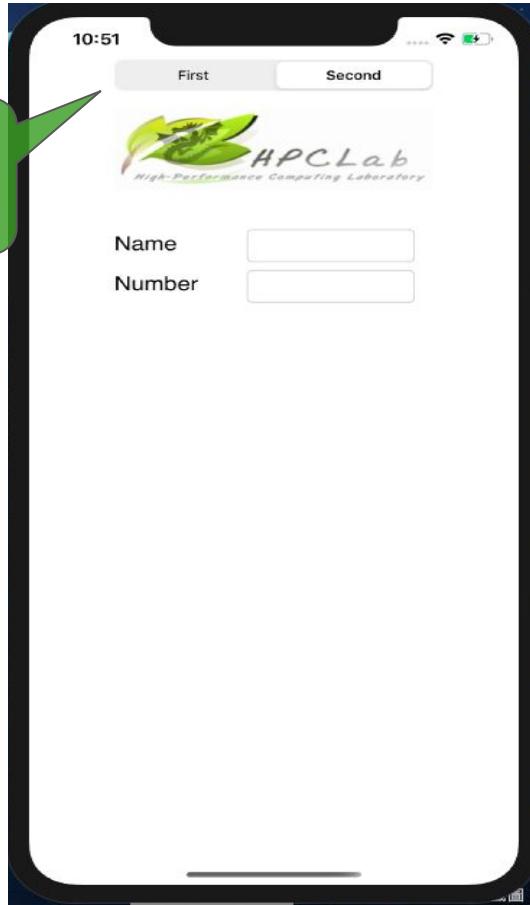
```
① @IBAction func secureText(_ sender: UISwitch) {  
36     if sender.isOn{  
37         numTextField.isSecureTextEntry = true  
38     } else {  
39         numTextField.isSecureTextEntry = false  
40     }  
41 }  
42 |  
43
```

當按紐設定為開時將數字隱藏

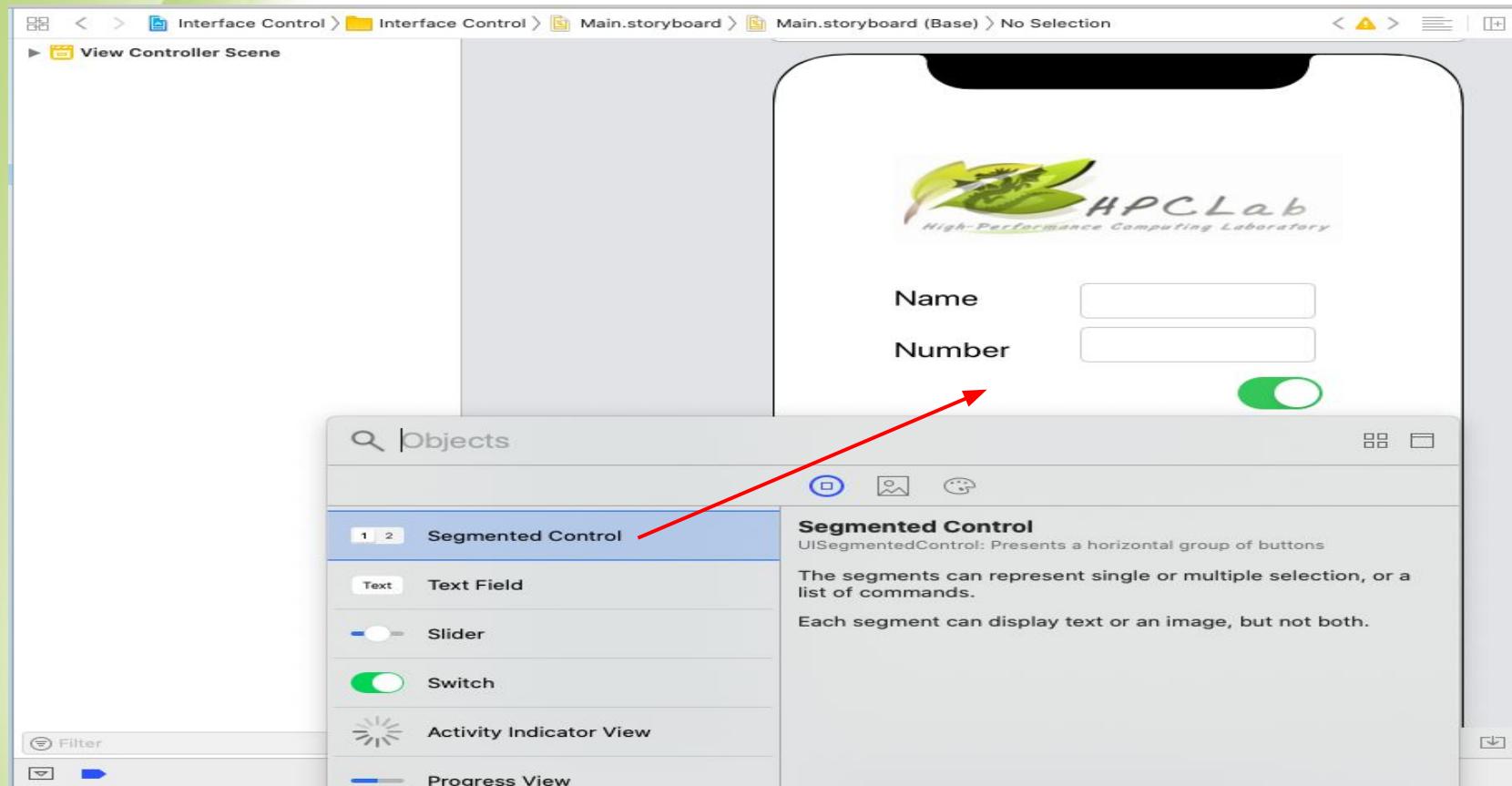
當按紐設定為關時顯示數字

Segmented Control

切換Segmented
來達到顯示或隱
藏其他元件



Segmented Control

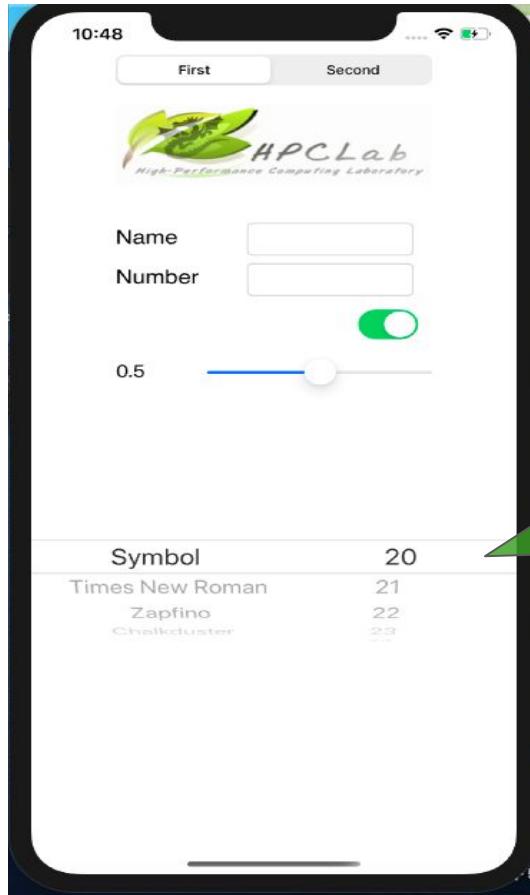


Segmented Control

```
Interface Control > Interface Control > ViewController.swift > ViewController
37 }
38
39 @IBAction func secureText(_ sender: UISwitch) {
40     if sender.isOn{
41         numTextField.isSecureTextEntry = true
42     } else {
43         numTextField.isSecureTextEntry = false
44     }
45 }
46
47 @IBAction func segmentCtrl(_ sender: UISegmentedControl) {
48     if sender.selectedSegmentIndex == 0{
49         slider.isHidden = false
50         `switch`.isHidden = false
51         alpha.isHidden = false
52     } else {
53         slider.isHidden = true
54         `switch`.isHidden = true
55         alpha.isHidden = true
56     }
57 }
58 }
```

當選擇第一頁時將元件顯示，選擇第二頁時隱藏元件

Picker View



左邊選擇
字型右邊
選擇文字
大小

Picker View

The screenshot shows the Xcode interface with a storyboard scene and the Objects library.

Storyboard Scene: The top half of the screen displays a storyboard scene titled "View Controller Scene". It contains a "View Controller" object which has a "Control" child object. The "Control" object contains several UI elements: "Safe Area", "HPC", "Name", "Number", "Num Text Field", "Name Text Field", "Slider", "Alpha", and "Switch". Below these are sections for "First, Second", "FirstResponder", "Exit", and "Storyboard Entry Point".

Objects Library: The bottom half of the screen shows the "Objects" library. A red arrow points from the "Picker View" section in the library to the "Picker View" component in the storyboard scene.

Objects Library Details:

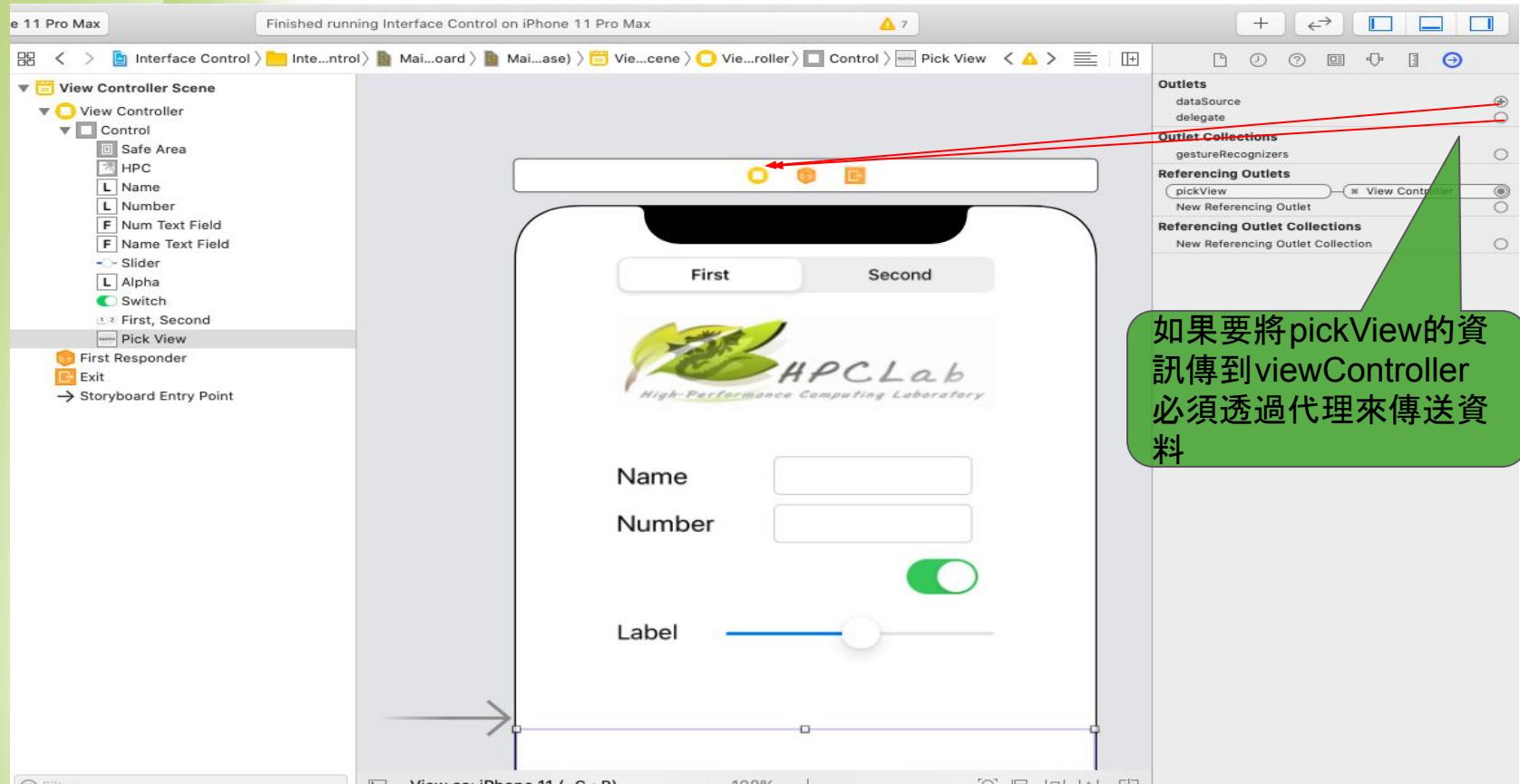
- Collection View**: Displays a grid of items.
- Collection View Cell**: A single item in a collection view.
- Collection Reusable View**: A reusable view for collection views.
- Text View**: A text input field.
- Scroll View**: A scrollable view.
- Date Picker**: A date selection component.
- Picker View**: The selected component, described as "Provides a potentially multidimensional user-interface element consisting of rows and components. A component is a wheel, which has a series of items (rows) at indexed locations on the wheel. Each row on a component has content, which is either a string or a view object such as a label or an image."

Picker View

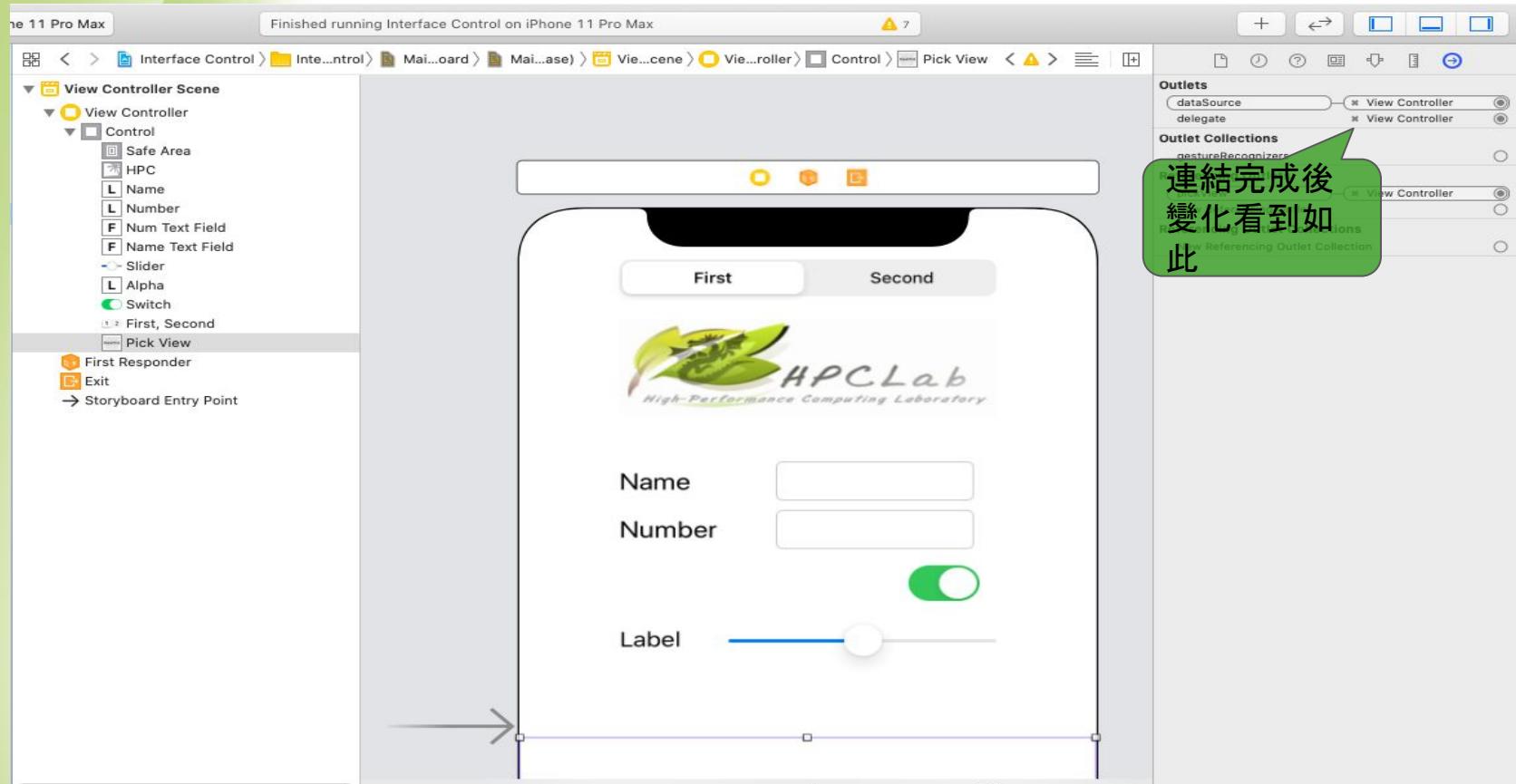
```
iPhone 11 Pro Max Finished running Interface Control on iPhone 11 Pro Max ⚠ 7
Interface Control > Interface Control > ViewController.swift > ViewController
import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var HPC: UIImageView!
14     @IBOutlet weak var nameTextField: UITextField!
15     @IBOutlet weak var numTextField: UITextField!
16
17     @IBOutlet weak var alpha: UILabel!
18     @IBOutlet weak var sliderValue: UISlider!
19
20     @IBOutlet weak var slider: UISlider!
21     @IBOutlet weak var `switch`: UISwitch!
22
23     @IBOutlet weak var pickView: UIPickerView!
```

pickView在
選擇字型與
文字大小時
無法傳送
ActionEvent
因此只能以
Outlet連結

Picker View



Picker View



Picker View

```
2 // ViewController.swift
3 // Interface Control
4 //
5 // Created by evan on 2020/3/3.
6 // Copyright © 2020 evan. All rights reserved.
7 //
8
9 import UIKit
10
11 class ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
12
13     @IBOutlet weak var HPC: UIImageView!
14     @IBOutlet weak var nameTextField: UITextField!
15     @IBOutlet weak var numTextField: UITextField!
16
17     @IBOutlet weak var alpha: UILabel!
18     @IBOutlet weak var sliderValue: UISlider!
19
20     @IBOutlet weak var slider: UISlider!
21     @IBOutlet weak var `switch`: UISwitch!
22
23     @IBOutlet weak var pickView: UIPickerView!
24
25
26
27     @IBAction func nameTextField(_ sender: UITextField) {
28         sender.resignFirstResponder()
29     }
30 }
```

viewController
必須要繼承
pickerView的
協定才能與
pickerView連結

Type 'ViewController' does not conform to protocol
'UIPickerViewDataSource'
Do you want to add protocol stubs? Fix

當我們繼承
完時會要求
必須要實作
協定的某些
函式，我們
可以按下 Fix
來看到要實
作哪些函式

Picker View

```
< > Interface Control > Interface Control > ViewController.swift > M numberOfRowsInSection(in:)

2 // ViewController.swift
3 // Interface Control
4 //
5 // Created by evan on 2020/3/3.
6 // Copyright © 2020 evan. All rights reserved.
7 //
8
9 import UIKit
10
11 class ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
12
13     func numberOfComponents(in pickerView: UIPickerView) -> Int {
14         code
15     }
16
17     func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int {
18         code
19     }
20
21
22     @IBOutlet weak var oneImageview: UIImageView!
23     @IBOutlet weak var oneTextField: UITextField!
24     @IBOutlet weak var numTextField: UITextField!
25
26     @IBOutlet weak var alpha: UILabel!
```

上面的函式代表有多少欄位

下面的函式是決定每一欄有幾個選項

Picker View

```
9 import UIKit
10
11 class ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
12
13     let fontName = ["Symbol", "Times New Roman", "Zapfino", "Chalkduster"]
14     let fontSize = [20, 21, 22, 23, 24, 25, 26]
15     var currentSize:CGFloat = 20.0
16
17     func numberOfComponents(in pickerView: UIPickerView) -> Int {
18         return 2
19     }
20
21     func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int {
22         if(component == 0){
23             return fontName.count
24         }
25         return fontSize.count
26     }
27
28 }
```

回傳2代表有兩欄的選項

先將每欄的選項以字串陣列宣告出來

component用來決定是哪一欄要回傳多少選項

如果是第一欄便回傳字型陣列有幾個元素

count來統計陣列有多少元素

Picker View

```
34
● @IBOutlet weak var HPC: UIImageView!
● @IBOutlet weak var nameTextField: UITextField!
● @IBOutlet weak var numTextField: UITextField!

38
● @IBOutlet weak var alpha: UILabel!
● @IBOutlet weak var sliderValue: UISlider!

41
● @IBOutlet weak var slider: UISlider!
● @IBOutlet weak var `switch`: UISwitch!

44
● @IBOutlet weak var pickView: UIPickerView

46
● @IBOutlet weak var name: UILabel!
● @IBOutlet weak var number: UILabel!

49
50
51
```

將Label連結以方便我們改它們的字型

Picker View

```
func pickerView(_ pickerView: UIPickerView, titleForRow row: Int, forComponent component: Int) -> String? {
    if(component == 0){
        return fontName[row]
    }
    return fontSize[row]
}
```

接著我們使用剛剛協定中的其他函式，這個函式用來顯示每個欄位的選項

第一列回字型的陣列，row是目前選到選項的index

Picker View

```
func pickerView(_ pickerView: UIPickerView, didSelectRow row: Int, inComponent component: Int) {  
    if component == 0 {  
        name.font = UIFont(name:fontName[row],size:currentSize)  
        number.font = UIFont(name:fontName[row],size:currentSize)  
    } else {  
        // 請自行在viewController宣告currentSize此變數  
        currentSize = CGFloat(Double(fontSize[row])!)  
        name.font = name.font.withSize(CGFloat(Double(fontSize[row])!))  
        number.font = number.font.withSize(CGFloat(Double(fontSize[row])!))  
    }  
}
```

這個函示用來決定當我們選到該選項要做的對應事情

更改第二欄選項時將字型大小修改並記錄修改完的大小

字串要轉CGFloat要先轉為Double

當更改第一欄的選項時將字型修改，因為不能只設定字型因此用一個變數來記錄文字大小