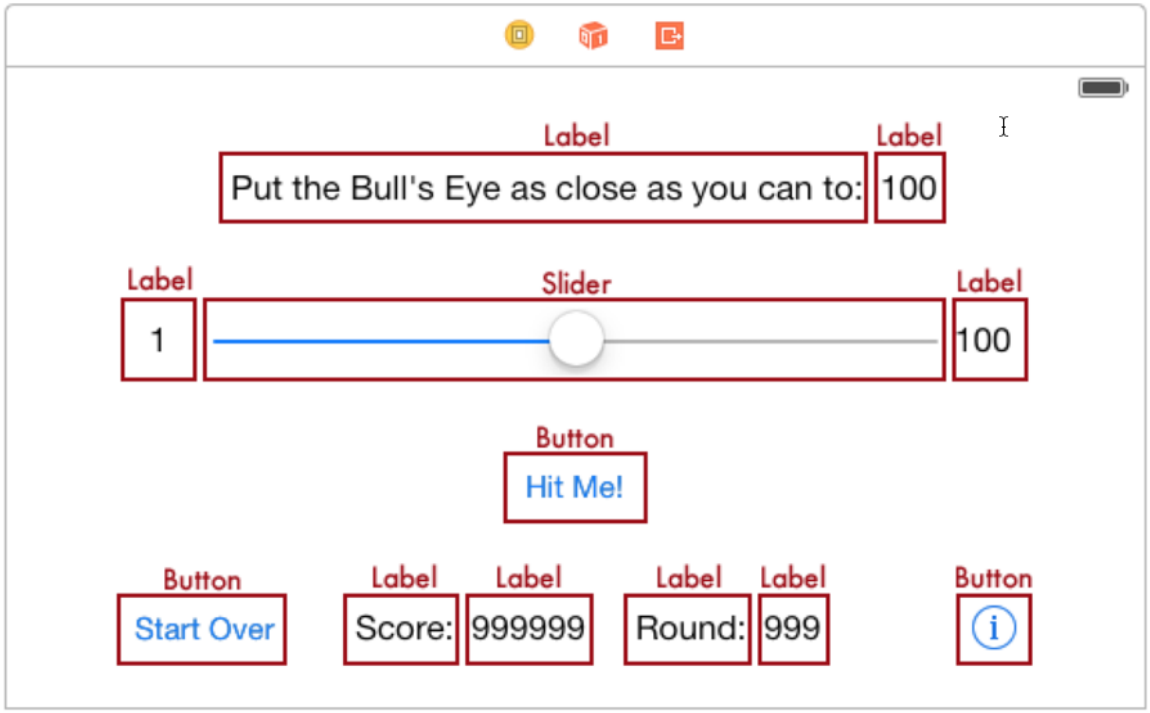
让不懂编程的人爱上iPhone开发(2017秋iOS11+Swift4+Xcode9版)-第4篇

休息的怎样了？是否已经迫不及待的想要继续新的学习了呢?

好吧，接下来我们就做点实际的事情。

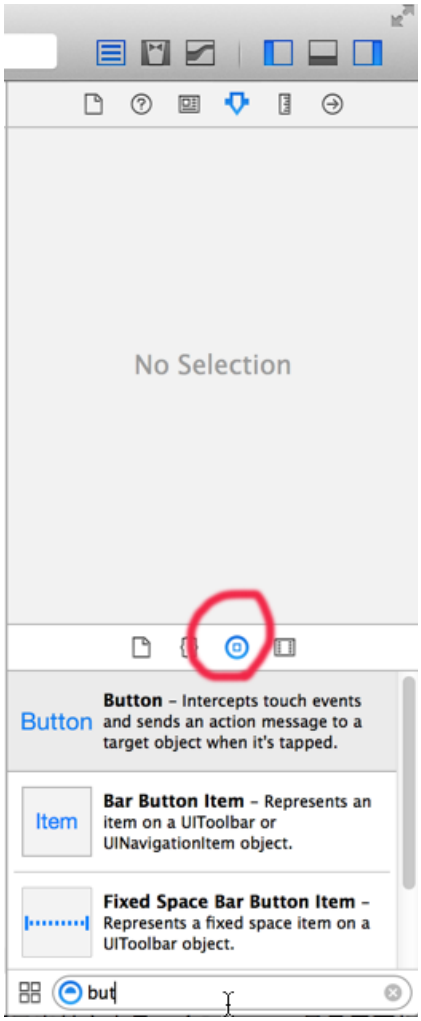
添加其它控件

到目前为止，我们的界面上只有以背景和一个按钮，接下来还是添加一些其它界面控件吧。下面是我们所创建的界面的最终效果。

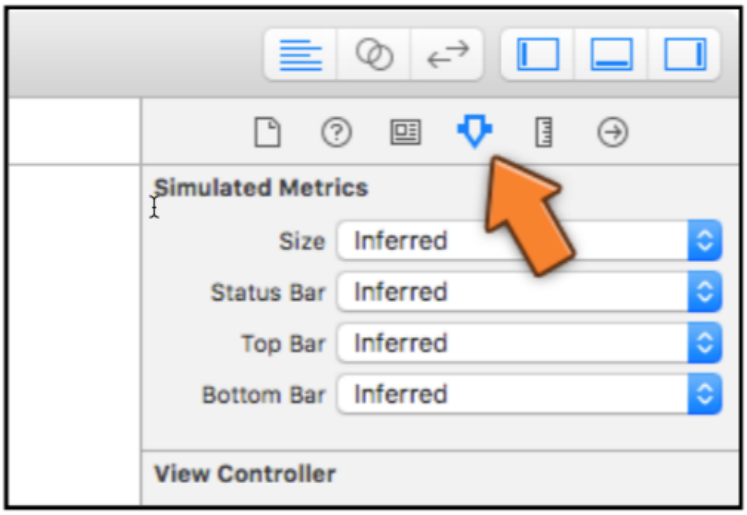


如你所见，我在部分标签处放了一些占位用的数值（比如999）。之所以这样做，是为了方便查看标签实际使用时在界面上的显示效果。玩家的得分可能会非常高（永远不要低估或高估玩家的智商~），因此最好预留足够的空间。

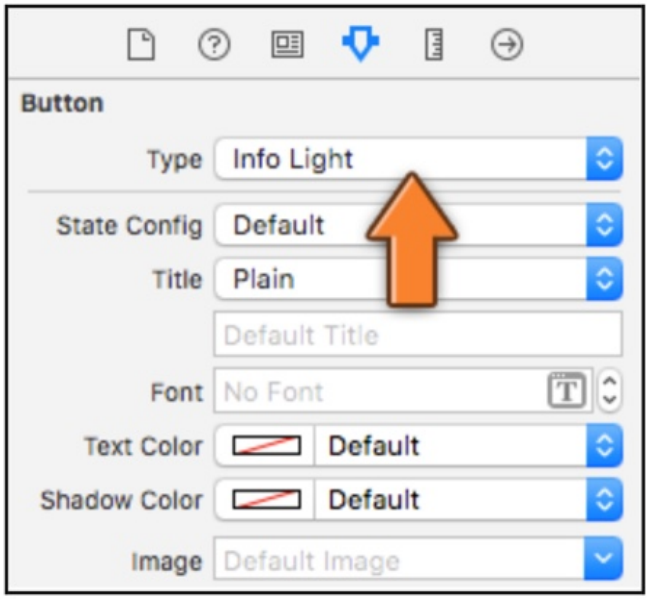
好了，现在该你自己出手了。如果你是个设计人员，相信你会喜欢下面的操作。打开Main.storyboard，尝试从对象库（Xcode右侧面板下面的Object Library）拖曳不同的控件放到视图上。也不用那么精确。实际上，要添加的三种界面元素是Label,Button和Slider，可以从Xcode界面右下的Object Library里找到这些元素.



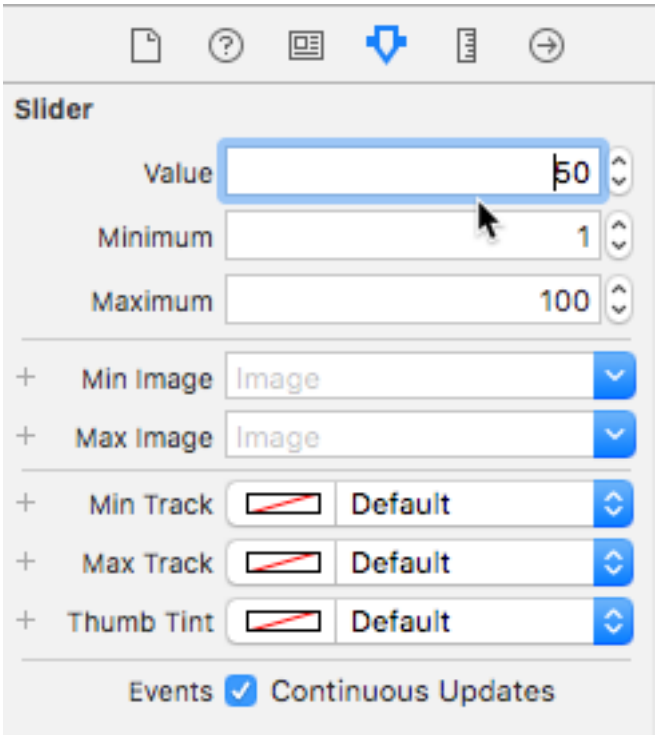
为了调整界面中UI元素的设置，我们需要用到所谓的Attributes inspector。我们可以在Xcode的右侧面板中找到该视图。



需要注意的是，类似”i”形状的界面元素其实也是一个Button，只是需要把它的类型设置为Info Light。



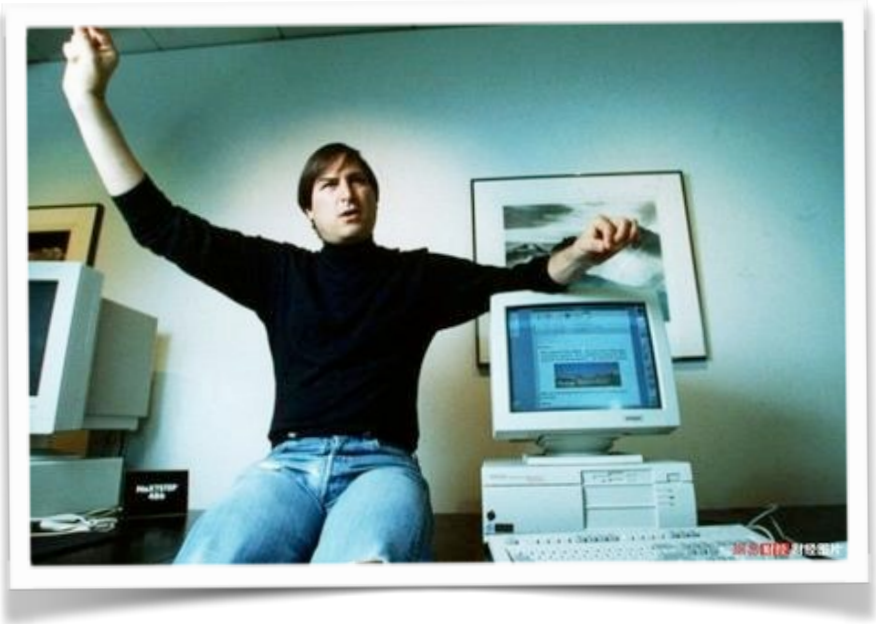
接下来让我们设置滑动条的数值。选中Slider，切换到Attributes Inspector，把它的最小数值设为1，最大数值设为100，当前数值设为50.



当你完成以上操作后，界面已经有了12个用户界面元素：1个滑动条，3个按钮，还有一堆标签。怎么样，很有成就感吧。

点击Run运行应用，然后好好玩上一会儿。除了之前的按钮，其它控件现在还做不了具体的事情，不过起码你可以拖着滑动条来回玩。

到目前为止，我们已经完成了界面的基本布局，而且不需要写一行代码。如果你是一个设计师，肯定要为此欢呼雀跃。可惜好日子到头了，很快我们就需要使用Swift编写代码让这些控件变得可交互。
乔帮主到目前为止对你的工作还比较满意，暂时没有召唤你去天国猛批一顿的意思。
不过按照他一向的完美标准，你还有很多工作要做。



让滑动条变得可交互

在我们to-do list上的下一个待办项目是：当玩家触碰按钮时读取滑动条上的数值。
如果你在Interface Builder的时候没有故意搞一些麻烦，比如装作不经意间把按钮和showAlert动作的关联取消，那么此时就可以更改代码让应用在弹出警告框中显示滑动条的数值。（如果你的确取消了按钮到动作的关联，那么首先需要再次将其关联起来。）

还记得如何在视图控制器中添加一个动作，从而让它可以识别用户对按钮的触碰吗？对于滑动条我们可以做同样的事情。一旦用户拖曳滑动条的手柄，就会触发这个动作。

要实现这一切，跟之前的操作几乎完全相同。

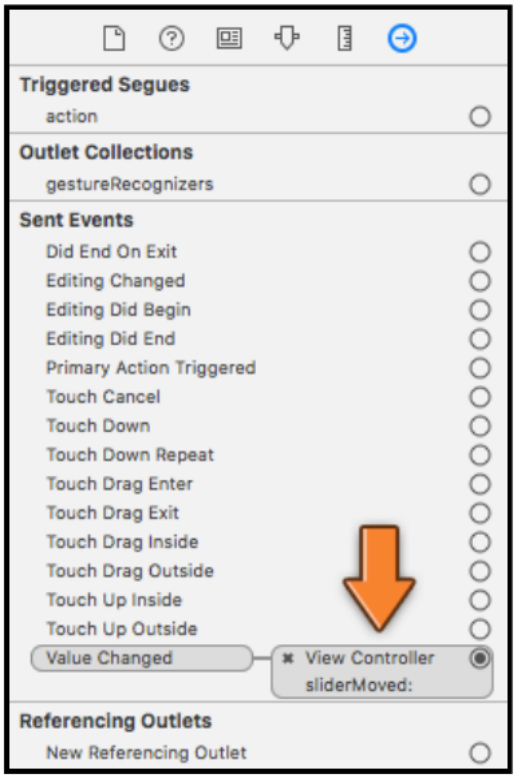
首先在Xcode中点击ViewController.swift，然后在最后一个花括号前面添加下面的代码：

```
    @IBAction func sliderMoved(slider: UISlider){
        print("滑动条的当前数值是： \(slider.value)")
    }
```

注意到此时@IBAction func sliderMoved(slider:UISlider)这行代码的左侧有个空心圆，代表它还没有跟storyboard中的界面元素关联起来。

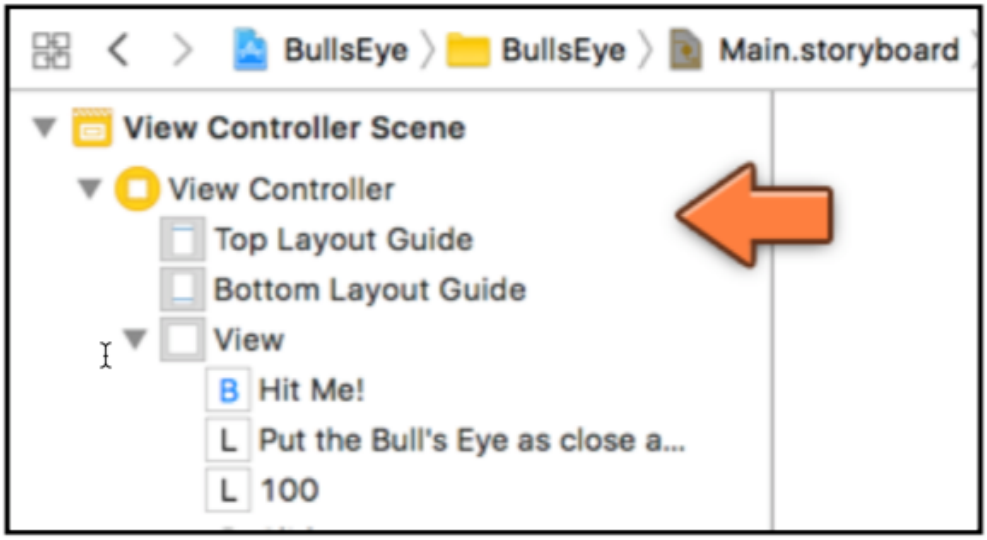
此时在Xcode中点击Main.storyboard，按住Ctrl键不放，点击鼠标左键从视图中的slider滑动条上拖一条线到对象面板(Outline pane）的View Controller上，然后从弹出菜单中选择sliderMoved:。

此时如果你通过Xcode右侧的面板切换到Connections inspector，就可以看到sliderMoved:动作和滑动条的Value Changed事件关联在一起。



这就意味着每当滑动条的数值发生变化时（用户拖动滑动条），就会调用sliderMoved()方法。

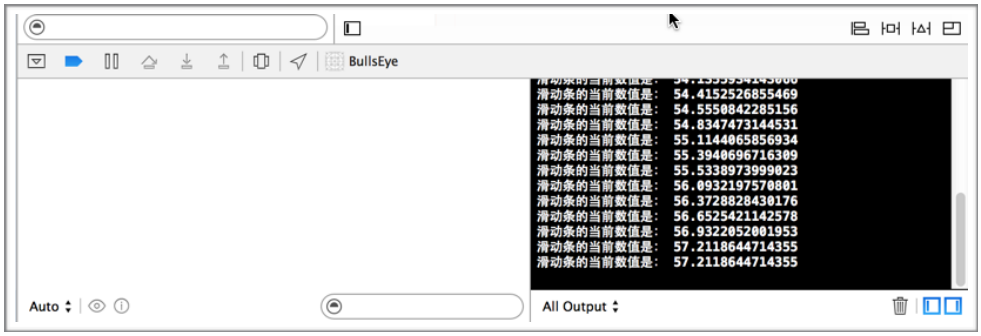
再次提醒大家，在Interface Builder的canvas左侧，是所谓的Document Outline，其中列出了当前视图中的所有视觉元素。



记住，如果你看不到Document Outline,可以点击Xcode窗口底部的小图标来显示：

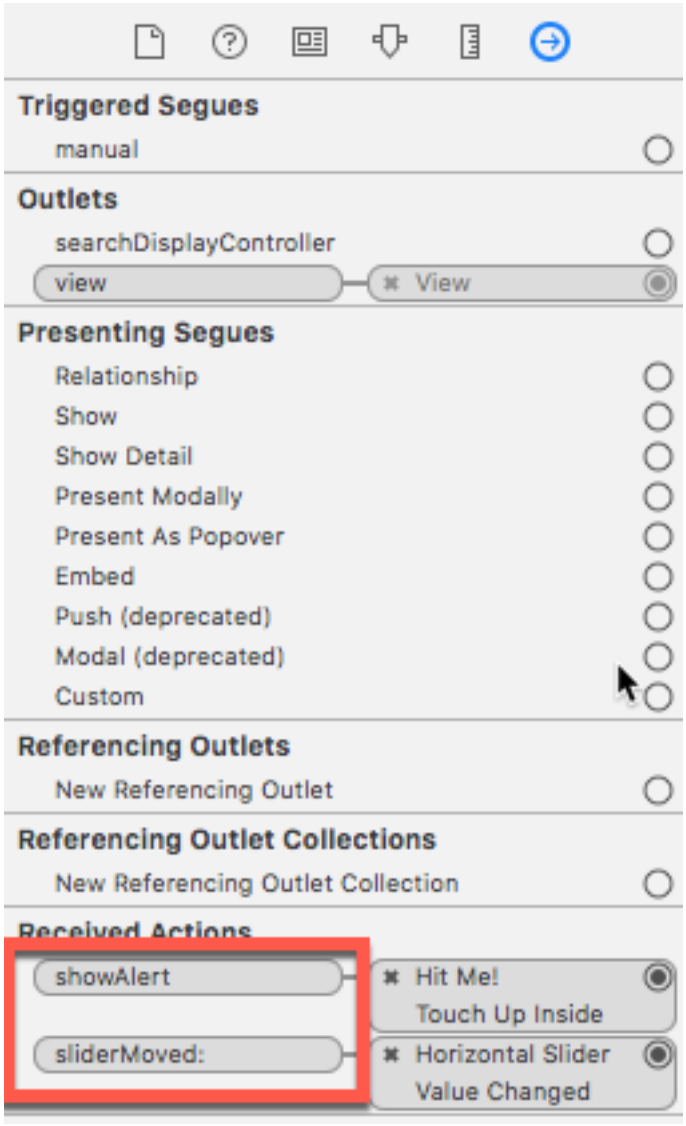
现在点击Run运行游戏，然后拖动滑动条看看反应。

一旦你开始拖动，Xcode窗口会在底部打开一个新面板，也就是传说中的Debug Area(调试区)，然后显示下面的信息：



如果你把滑动条拖到最左边，会看到上面显示的数值变成1.0，如果拖到最右边，那么显示的数值变成100.0.

print()函数可以帮忙我们了解应用的逻辑是否正常。它的作用就是在Debug调试区显示一条文本信息。这里我们用它来验证滑动条是否和指定的动作关联在一起。在我每次要添加新的功能前，我都会用print()来确保之前一切都OK.



提醒：

你是否注意到sliderMoved:方法的名称后面有一个冒号，而showAlert却没有？这是因为sliderMoved:方法有一个参数slider,而showAlert则没有任何参数。如果某个动作方法有一个参数，那么Interface Builder就会在名称后面添加一个冒号。很快我们就将了解更多关于参数的问题。

科普时间

好了，马上又要到科普时间了。这样免得你太累，不过如果还是那句话，如果你对理论知识无爱，可以跳过去无视。

首先我们来科普几个东西，所谓的iOS开发，App Store,Mac开发，Xcode，Objective-C, Swift,Cocoa, Cocoa Touch究竟是什么关系。

在我初学苹果开发的时候，经常把这些东西搞混，因为早期的各种编程书籍中既有iOS开发，又有Objective-C开发，还有Cocoa 开发。

程序猿最NB之处，同时也是最让人讨厌的地方就是，喜欢用各种术语，各种缩写让你觉得自己是个白痴。虽然我们的目标不是成为最NB的程序猿，但了解一些相关的开发术语没有坏处。

我并不指望你一下子就看懂它们的真正用处和区别，但起码先留下点印象，然后在后面的教程中再逐步熟悉。对于教程中的其它抽象概念，哥也是类似的做法，先简单介绍，然后让你反复接触，直到彻底进入你的盗梦空间。

iOS开发，在2010年推出iPad之前其实就是iPhone开发。所以很多早期的iOS教程都写的是iPhone应用开发。大家都知道2007年macworld上帮主的那次惊天地泣鬼神的神级演讲，iPhone的首次展示让我这个果粉恨不得立马换国籍。

2008年前第三方只允许开发Safari上的网页应用。2008年开始，苹果在当时的SVP Scott Forstall的带领下向开发者正式推出了iPhone SDK，并直接打造了一个完整的生态系统。

2010年帮主发布了耶稣之本iPad，同年的WWDC上将iPhone OS更名为iOS。iOS基于Mac OS X系统开发，但针对移动设备特有的硬件特性做了大量的改善和优化。

如今的iOS开发泛指针对所有安装了iOS操作系统的设备（当然只限苹果生产）开发应用或游戏。主要包括：iPhone全系列，iPod touch全系列，iPad全系列。

目前除了iOS,还有了针对Apple Watch的Watch OS，还有针对Apple TV的TV OS，不过它们都是基于iOS衍生而来的~

App Store，顾名思义就是苹果卖针对iOS设备上应用和游戏的软件商城。

Mac开发，指的是开发Mac操作系统下的应用和游戏软件。在iOS和App Store取得了巨大的成功后，苹果把iOS的一些成功特性开始反哺给Mac操作系统，同时在2011年推出Mac版的 App Store。
不过目前看来Mac开发并没有吸引足够多的开发者。也没有多少非常成功的案例。

Cocoa和Cocoa Touch上一次的内容中提过，同样是编程环境，一个用于Mac开发，一个用于iOS开发。

Swift和Objective-C属于编程语言，和C,C++,Java,C#,Javascript,PHP,Python,Ruby等相似。

Xcode是Mac 平台下的软件开发环境，可以开发Mac和iOS应用。

如果和其它开发平台做一下对比（可耻啊。。。），可能很多人就明白了。

Xcode类似于Visual Studio或者Eclipse

Swift和Objective-C类似于C,C++,Java,C#这些开发语言，

Cocoa 和Cocoa Touch 类似于微软开发中的MFC或.NET.

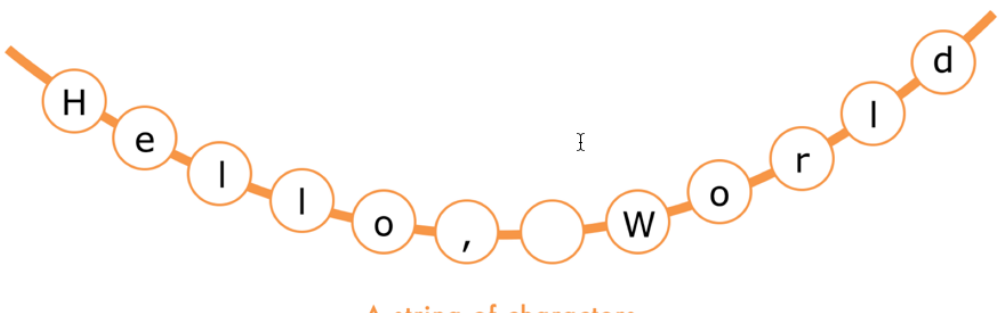
Swift/Objective-C和Cocoa/Cocoa Touch的关系类似于C++和MFC,或者C#和.NET的关系。

继续科普-什么是字符串

在刚才的print()那行代码里，我们用到了这样的一个东西，
"滑动条的当前数值是： \(slider.value)"

这就是个字符串，到目前为止我们已经用了好几个类似的东西。比如UIAlertController（提示对话框）里面的就是字符串。

通常来说，这样一连串的文字被成为字符串，因为我们可以把文字看做字母，数字，标点符号的一个序列，就好像用串在一起的珠子。



在我们开发应用的过程中，将会大量使用字符串，所以很快你就会熟悉它的用法。

在Swift里面，为了创建一个字符串，只需要把文字放到双引号里面就好了。这个@符号很重要！如果你之前用其它的编程语言写过代码，或许在生成字符串的时候可以使用双引号或单引号（比如python)，但是Swift只能使用双引号。

而且很重要的一点是你必须使用半角输入，不要使用中文输入法的全角双引号。

总结一下：

// 在Swift 中正确使用字符串的方法:
"I am a good string"

// 下面都是错的:
'I should have double quotes'
"Two single quotes do not make a double quote''
“My quotes are too fancy”
@"I am an Objective-C string"

在print()这行代码里面，用到的字符串是"滑动条的当前数值是： \(slider.value)”

所有在符号 \(…)之间的字符串都是特殊的占位符。

比如下面的这个占位符："滑动条的当前数值是：: X”，这里的X将被滑动条的数值所替代。

让变量来帮忙

在调试面板中用print()打印信息对于开发和测试非常有用，不过玩家对这种消息可是毫无兴趣。因此让我们来改进一下这个动作方法，让它在一个提示对话框里面显示滑动条的数值。那么我们该如何把滑动条的数值送给showAlert()呢?

当我们在sliderMoved()中读取滑动条的数值时，一旦这个动作方法结束，这个数据信息也就丢失了。我们需要记住这个数据，直到玩家触碰了按钮为止。

幸运的是，Swift（其它语言也是）为我们提供了一个很好的工具-变量。

在Xcode中打开ViewController.swift，在class ViewController这行代码的下面添加一行代码：

var currentValue: Int = 0

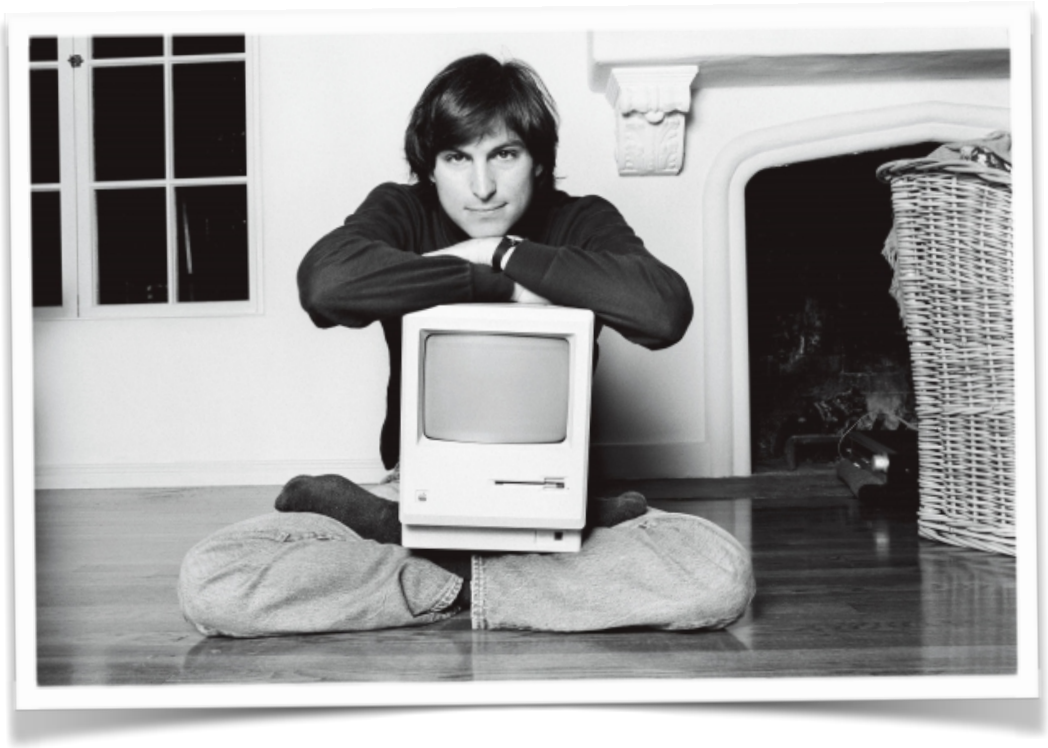
添加完成后的代码如下：

```
import UIKit

class ViewController: UIViewController {

  var currentValue: Int = 0

  override func viewDidLoad() {
    …
  }

  override func didReceiveMemoryWarning() {
     …

  }


  @IBAction func showAlert(){
    …

  }

  @IBAction func sliderMoved(slider: UISlider){
     …
  }
}
```

注意：上面唯一使用黄色高亮的代码才是刚刚新添加的代码，其它一切都保持不变。省略号…其实就是之前的代码内容，千万不要以为哥把那些代码都删掉了。如果你不确定的话，最简单的方式就是参考本章的参考项目源代码，对比下就知道了~
现在我们就添加了一个名为currentValue的变量到视图控制器中。变量被添加到方法的上面，通常我们需要让这行代码缩进显示，使用tab键或者用空格键。

至于使用两个空格还是4个空格取决于你的个人习惯，我们可以通过Xcode的偏好设置来设置这一点。在Xcode的顶部菜单中点击Xcode -Preferences..-Text Editing，然后跳转到Indentation选项卡即可。

还记得帮主当年的话吗？由内到外都要美。哪怕是代码的缩进显示这样小的细节，以后也会给你节省很多时间，让你读代码没那么累。更重要的是，真正的美，由内到外。很多偷懒的程序猿在这方面都过于随意，我毫不奇怪他们的代码里面经常会出现各种bug。这个无关技术，和心态有关。用心做，才能成为食神；用心演，才能成为喜剧之王~很多事情谈不上需要多高深的技巧，谈不上拼爹和干爹，谈不上苦逼和潜规则，唯用心与否而已。



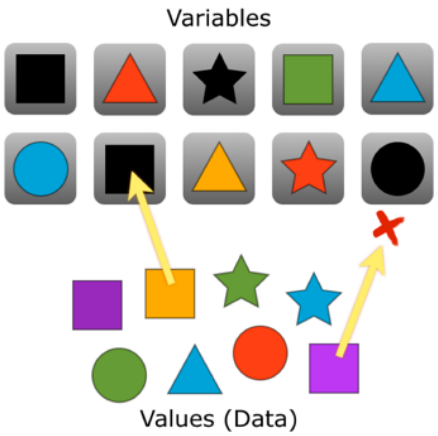
在之前的教程中曾提过视图控制器，或是任何一个对象都有自己的数据和功能。

showAlert()和sliderMoved()动作就是功能的典型例子，而currentValue变量则是数据的一部分。

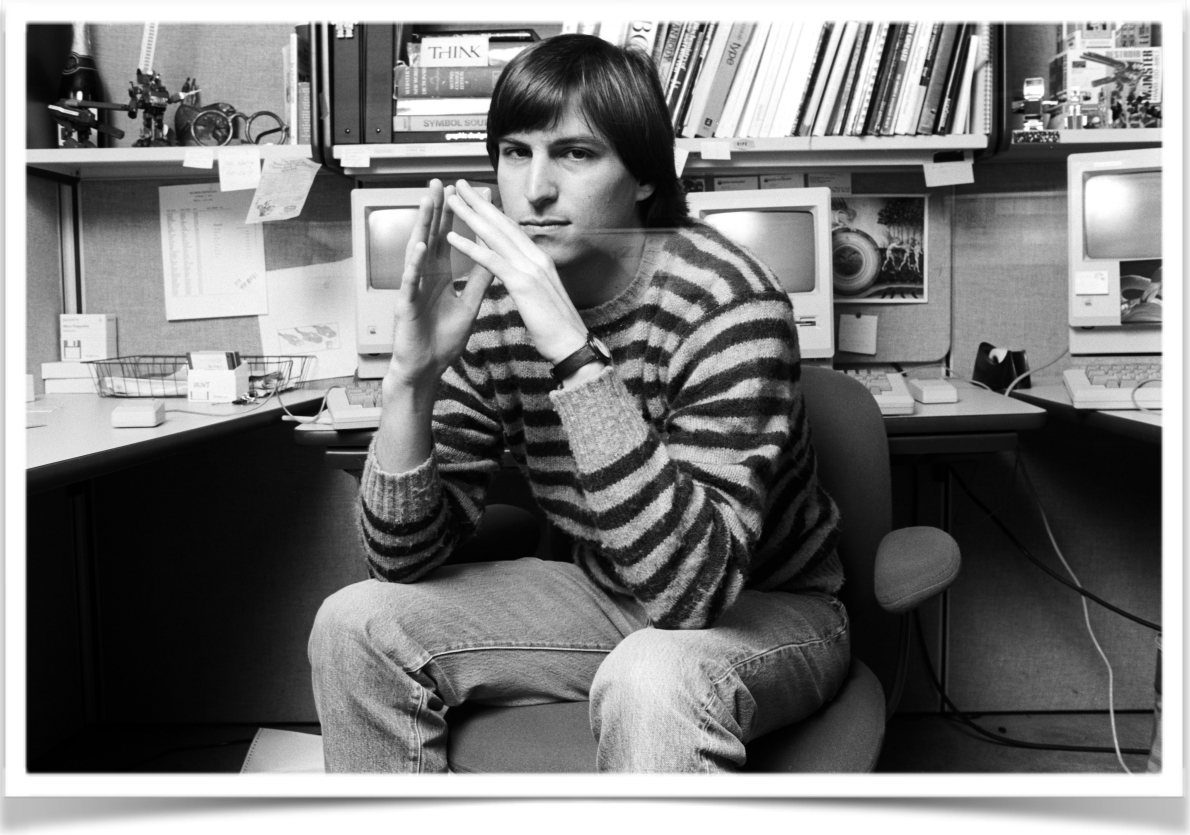
通过使用变量，可以让我们的应用拥有记忆。你可以把变量看做是存储某个数据的临时储物箱。正如储物箱有各种类型和尺寸的一样，数据也五花八门。

你不能把东西扔到储物箱里面然后撒手不管，因为经常会放入一些新的东西。当你的应用需要记住一些变化时，就需要把旧的数据拿出来，然后把新的数据放进去。

这就是变量(variable)的本质-变(vary)。比如说，每次玩家拖动滑动条的时候，我们都会使用滑动条的当前位置来更新currentValue。



储物箱的大小和变量可以保存的数值种类由datatype（数据类型）决定。这里我们指定currentValue这个变量的数据类型为Int（也就是integer），意味着储物箱里面可以放入整数（又称为integer），范围在正负20亿之间。Int是最经常用到的数据类型，不过很快我们会接触到其它的类型。



变量就象小孩的玩具积木一样：

我们需要把正确的形状放到正确的储物箱里面。储物箱就是变量，而它的数据类型（datatype）决定了里面能放什么形状的东西。形状就是你可以放入变量的可能数值。我们可以随后更改每个箱子里面的内容，可以拿出蓝色的方块积木，放进红色的方块积木，但前提是它们都是方块形状的。你不能把方块积木放到一个圆孔里面去：数值的数据类型和变量的数据类型必须是匹配的。
刚才也说了，变量是一个临时的储物箱。既然是临时，那么能保存多久呢？每个变量都有自己的生命周期（或者叫scope，术语狗滚粗），它的生命长短取决于你在程序的哪个位置定义变量。在这里，currentValue的声明和它的拥有者一样长，它的拥有者是ViewController。它们的命运交织在一起。在这里，只要我们不退出应用，这个视图控制器，还有currentValue就会活着。当然，很快我们就会了解到寿命很短的变量。
在程序这样一个虚拟世界中，每个变量，每个视图控制器都是一个虚拟的生命。世界毁了，所有的生命都会消亡。世界没有毁，有的生命也会消亡。这方面似乎可以和人类与宇宙的关系类比。

八卦时间：

刚才在谈到变量的时候，举了个例子，说是’不能把方块积木放到一个圆孔里面去’，英文原文是，you can’t put a square in a round hole

这让我想到了苹果著名的品牌广告Think Different的台词，人跟变量终究还是不同的。

变量不能做到随心所欲不逾矩，而人需要与众不同。

Think Different

Here's to the crazy ones.献给狂放不羁的一群人
The misfits. 他们是：不和主流的怪才
The rebels. 叛逆传统的勇士
The troublemakers. 制造麻烦的一笑撮
The round pegs in the square holes. 方凿圆枘、特立独行
The ones who see things differently. 他们观察问题与众不同
They are not fond of rules. 他们不喜欢条条框框
And they have no respect for the status quo. 更不把正统放在眼里
You can quote them, 你可以引用他们，
disagree with them, 也可以否决他们，

glorify or vilify them. 赞扬或是诋毁他们
About the only thing you can't do, 但只有一件事你不能做
is ignore them. 那就是漠视他们
Because they change things. 因为他们改变了事物
They push the human race forward. 他们推动了人类的进程
And while some see them as the crazy ones, 虽然有些人把他们当作疯子
We see genius. 但我们看见的却是天才
Because the people who are crazy enough to think 因为只有那些足够疯狂认为
they can change the world, 可以改变世界的人
Are the ones who do. 才能真正做到这一点