让不懂编程的人爱上iPhone开发(2013秋iOS7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。

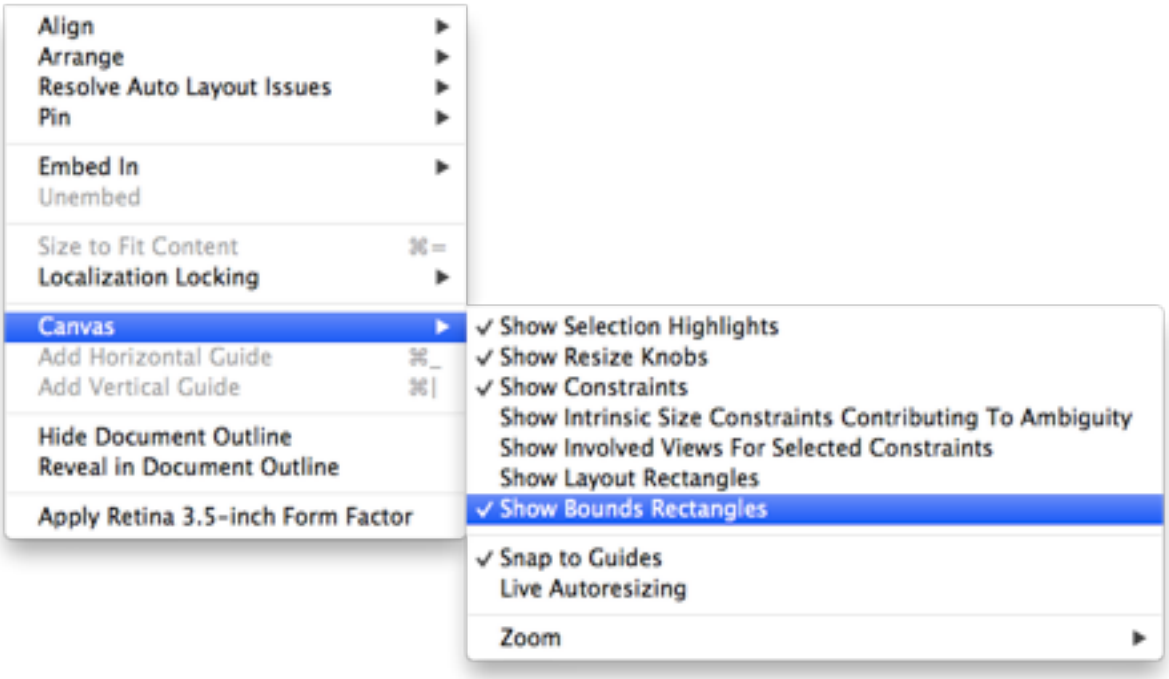
添加其它控件

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。

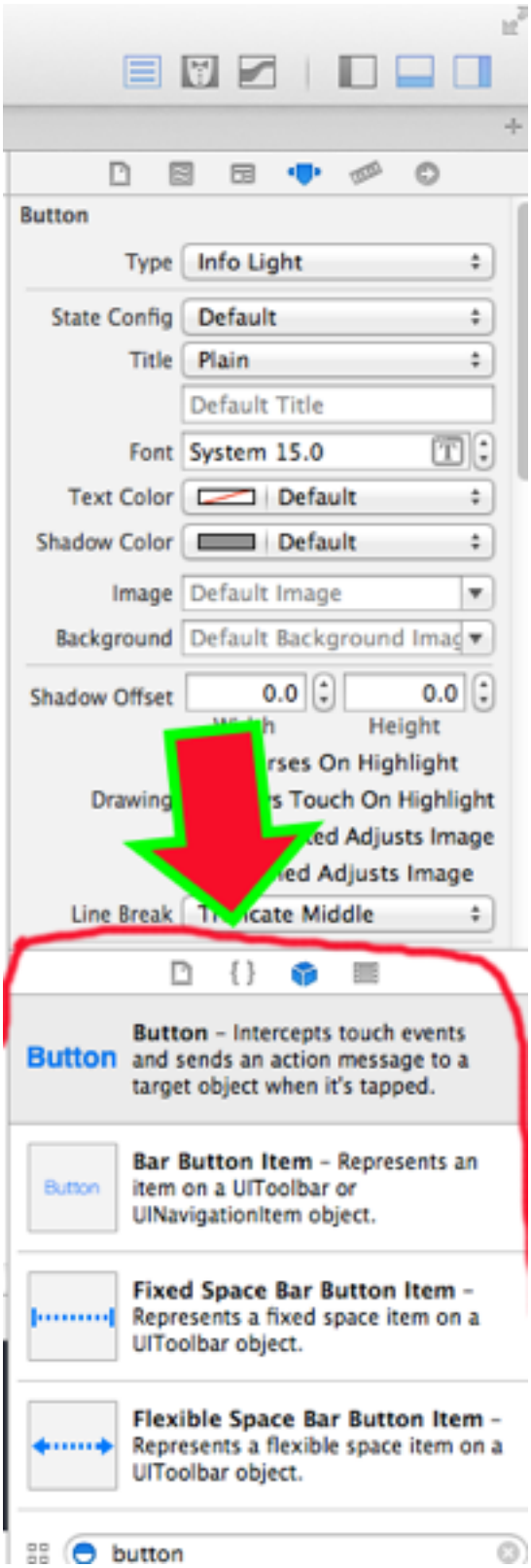
如你所见，我在部分标签处放了一些占位用的数值（比如499）。之所以这样做，是为了方便查看标签实际使用时在界面上的显示效果。玩家的得分可能会非常高，因此最好预留足够的空间。

另外你会看到，在每个界面元素上都有一个小框，但是默认好像没有啊？怎么来的呢？

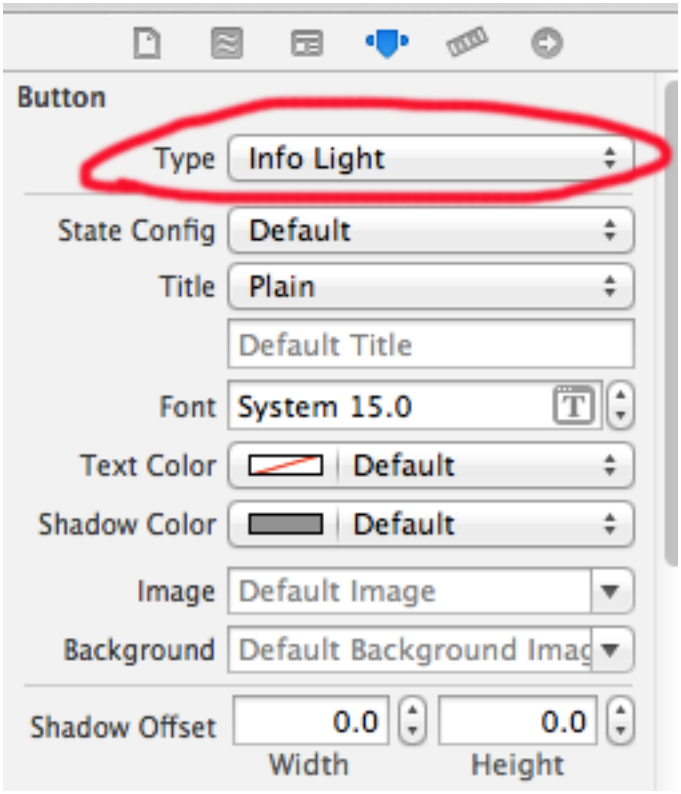
很简单，点击Xcode上面的菜单栏，选择Editor-->Canvas--->Show Bounds Rectangles。这样就可以看到每个界面元素的边界了。当然，这个只是为了方便开发者使用，实际调试或在设备上运行的时候是看不到外框的。



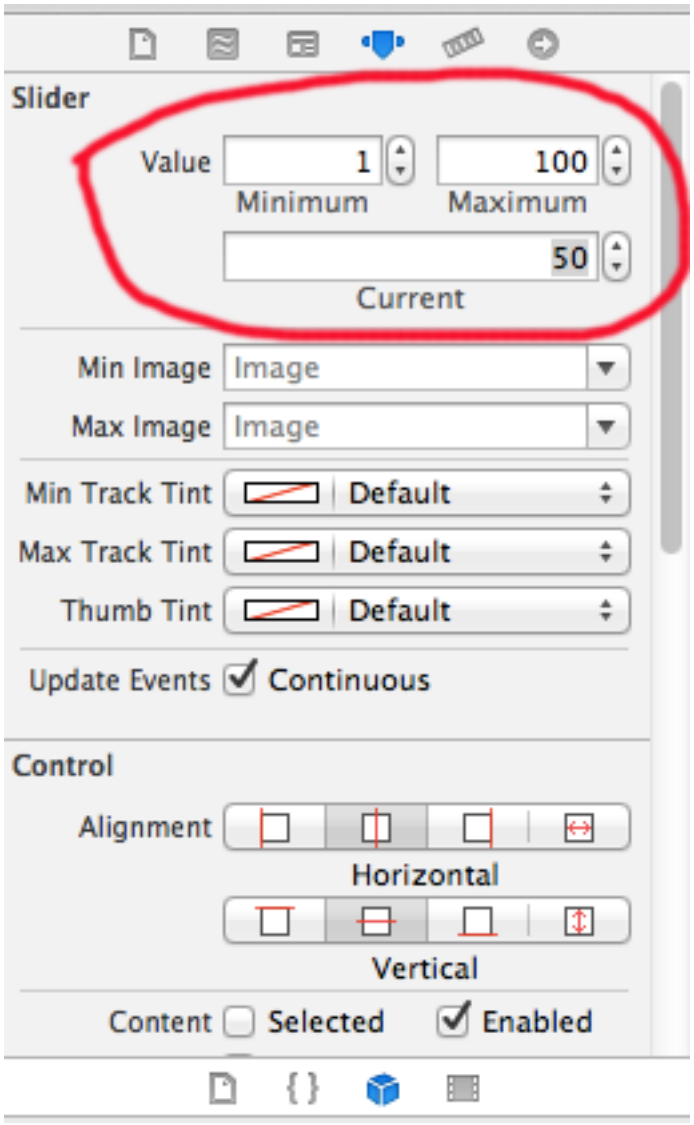
好了，现在该你自己出手了。如果你是个设计人员，相信你会喜欢下面的操作。打开Main.storyboard，尝试从对象库（Xcode右侧面板下面）拖曳不同的控件放到视图上。也不用那么精确。实际上，要添加的三种界面元素是Label,Button和Slider，可以从Xcode界面右下的Object Library里找到这些元素.


Button
Type
Info Light
State Config
Default
Title
Plain
Default Title
Font
System 15.0
Text Color
Default
Shadow Color
Default
Image
Default Image
Background
Default Background Imag
Shadow Offset
0.0
0.0
Height
Drawing
Line Break
Button
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Button
Bar Button Item – Represents an item on a UIToolbar or UINavigationItem object.
Fixed Space Bar Button Item – Represents a fixed space item on a UIToolbar object.
Flexible Space Bar Button Item – Represents a flexible space item on a UIToolbar object.
button

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。

当然，如果此时你把模拟器更改为iPhone Retina (3.5-inch)，可能会发现右边的部分视觉元素看不到。这个功能使用Auto Layout 功能很容易解决，不过在这里我们暂时不打算讨论这个话题。

关于xib和nib

如果这不是你读过的第一个iOS开发教程，也可能你听说过nib文件。从技术的角度讲，xib文件将会被编译成一个nib文件，然后放到应用束中。Nib这个术语主要是因为历史原因而存在。作为一个普通程序猿，你可以把xib和nib看成一回事。因为人们习惯称nib，所以可能后面我也会这么说。

如果你对历史八卦感兴趣，这里可以简单提提， 不感兴趣的话可以跳过这部分~

首先nib的三个字母分别代表NeXT(帮主回到苹果前搞的操作系统，后来演化成如今的Mac和iOS)，ib其实是Interface Builder的简写。
下面这图是1993年的乔帮主，风华正茂。

我们都知道Interface Builder提供了一系列的用户界面对象，比如上面用到的按钮，标签，滑动条之类的控件。开发者只需要把控件拖到视图中就可以完成界面的设计（产品经理和设计人员的大爱，我猜？），然后，用连线把各种action(动作)，控件对象和代码中的方法(method),接口（outlet）连接起来，就完成了创建工作。和Visual Studio这样的可视化编程环境相比，Xcode的这种开发方式大大减少了MVC模式中控制器和视图的耦合。在代码中（.h和.m文件），IBAction标记接受动作的方法，而IBOutlet标记控件对象的接口。在编译的过程中，Xcode会把xib中所有的用户交互元素对象序列化为XML文件或NeXT风格的属性列表文件,从而编译成一个nib文件放到应用束里。在应用程序运行的时候，会把nib里面的对象调入内存，并和二进制代码联系起来。
具体参考：
http://zh.wikipedia.org/wiki/Interface_Builder

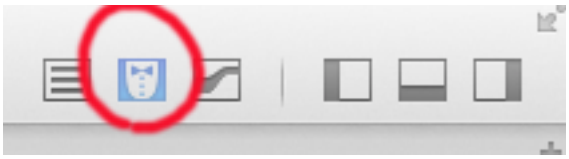
好了，八卦结束，让我们继续前进。

让滑动条变得可交互

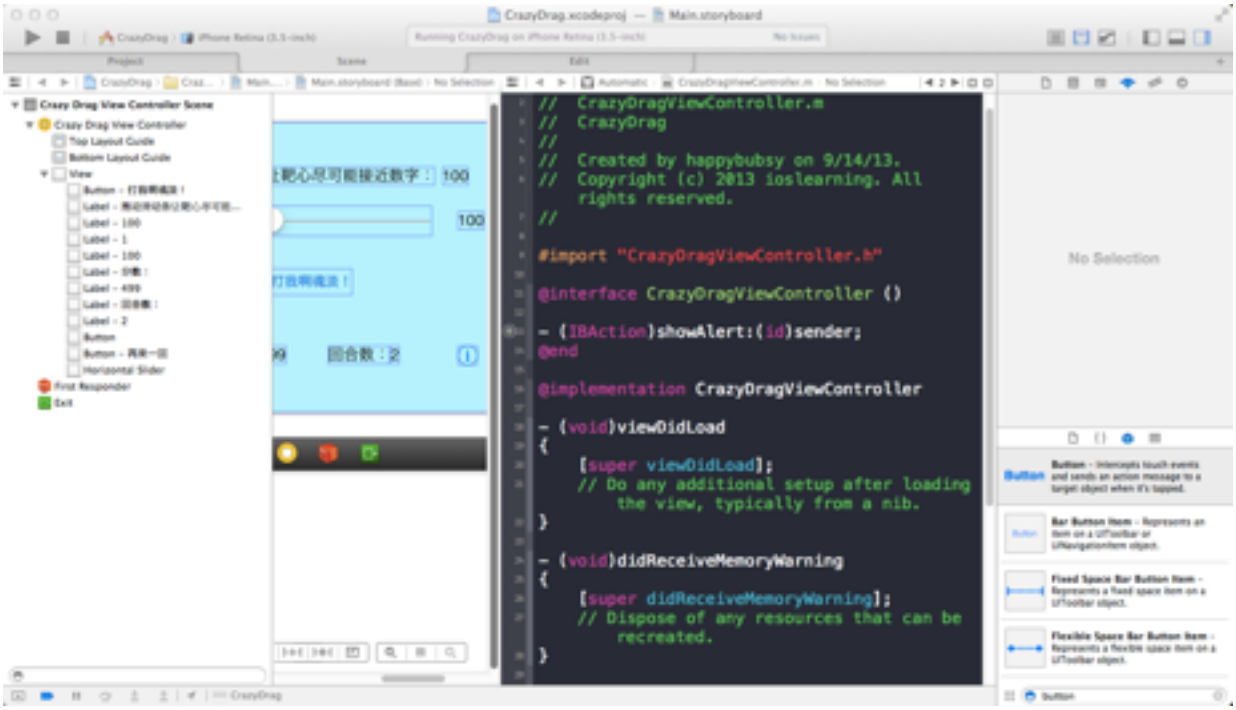
我们to-do list上的下一个项目是：当玩家触碰按钮时读取滑动条上的数值。

还记得我们怎么把一个动作添加到视图控制器上，从而让它可以知道玩家何时触碰按钮的吗？对于滑动条我们可以做同样的事情。每当玩家拖动滑动条时，就会执行这个动作。添加这个动作的操作步骤和之前几乎一模一样。

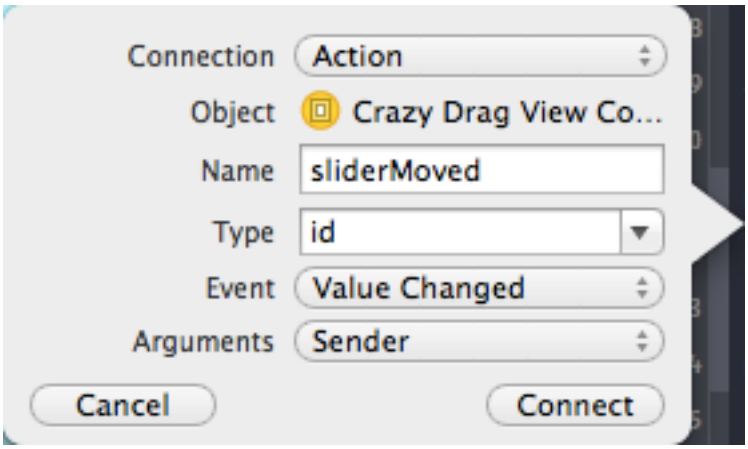
首先在Xcode中点击Main.storyboard，点击Xcode右上的show assistant editor。



此时的Xcode界面会变成下面的样子：



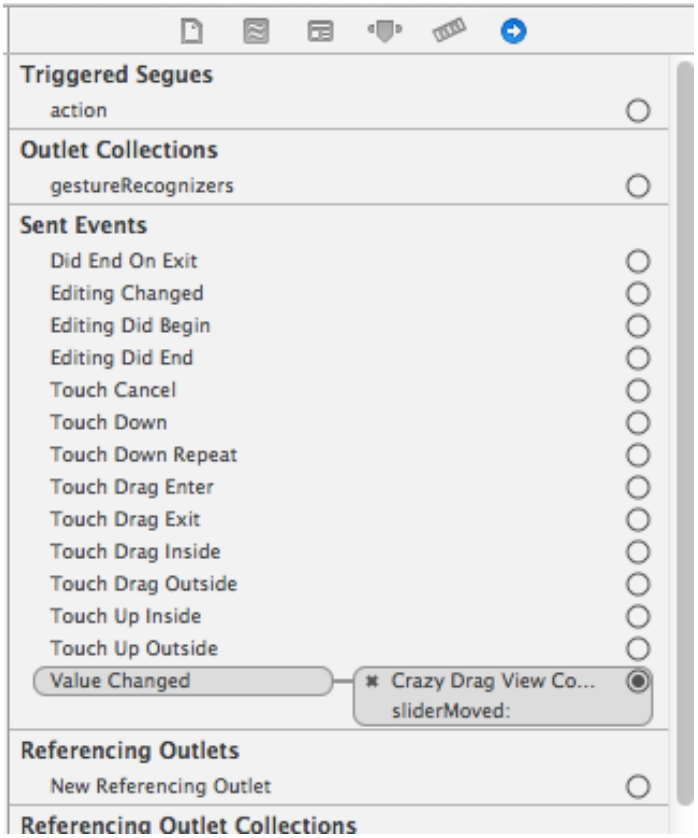
选中滑动条（Horizaontal Slider），按住control键，有鼠标拖出一条线到花花绿绿的代码区的@interface和@end之间，这时会弹出一个小的提示框，可以按照下图修改其中的内容，把Connection后面的选项选择Action，在Name后面输入sliderMoved，然后点击Connect。



这个时候你会看到在@interface和@end之间出现一行新的代码：

- (IBAction)sliderMoved:(id)sender;

这个时候查看右侧的Connections Inspector，你会发现sliderMoved动作和滑动条的Value Changed事件关联在一起了。这就意味着每当滑动条的数值发生变化时（也就是玩家来回拖动滑动条的时候），都会执行该动作。



当然，现在我们只是告诉滑动条有这样一个动作存在，至于这个动作会做什么，稍后才会具体来定义。

切换到CrazyDragViewController.m，在@implementation下面找到这段代码：

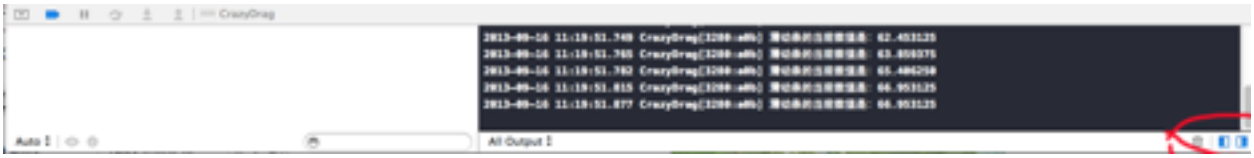
```
- (IBAction)sliderMoved:(id)sender {
}
```

把这段代码的内容修改为：

```
- (IBAction)sliderMoved:(id)sender {
    UISlider *slider = (UISlider*)sender;
    NSLog(@"滑动条的当前数值是：%f",slider.value);
}
```

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：



需要注意的是，如果你看不到这些信息，要检查下上图右下角红色标出部分是否选中。

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，如果拖到最右边，那么显示的数值变成100.

NSLog可以说是iOS开发中最强大的调试辅助工具。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NSLog来确保之前一切都OK.

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，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。

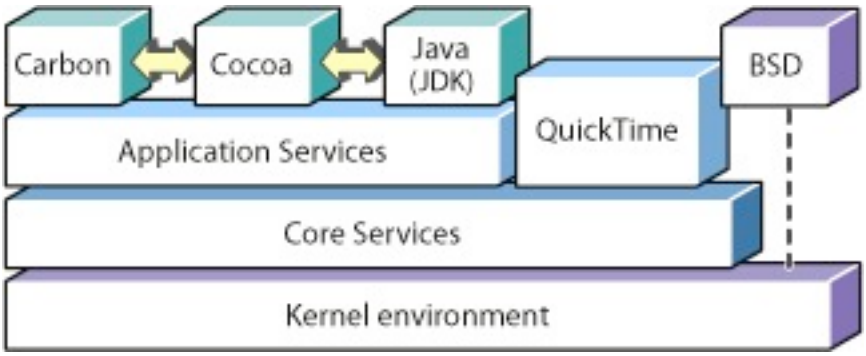
如今的iOS开发泛指针对所有安装了iOS操作系统的设备（当然只限苹果生产）开发应用或游戏。主要包括：iPhone全系列，iPod touch全系列，iPad全系列。

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 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。
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。

如果和微软平台做一下对比（可耻啊。。。），可能很多人就明白了。
Xcode类似于Visual Studio
Objective-C类似于C,C++,C#这些开发语言，
Cocoa 和Cocoa Touch 类似于微软开发中的MFC或.NET.
Objective-C和Cocoa/Cocoa Touch的关系类似于C++和MFC,或者C#和.NET的关系。

苹果官方文档的一个图很好说明了Cocoa在系统架构中的位置，看看就好，不必深究。

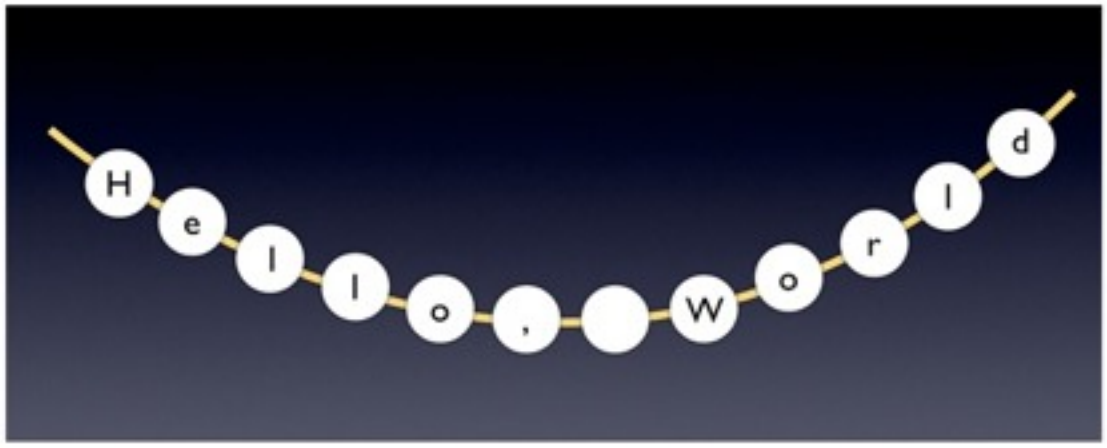


继续科普-什么是字符串

在刚才的NSLog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：%f"

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UIAlertView（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Objective-C里面，为了创建一个字符串，只需要把文字放到双引号里面，然后加上一个@前缀就好了。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不需要这个@符号，只需要双引号或单引号。当然，在Objective-C中也有可能不用@就生成字符串，不过那样就没法让你获取想要的字符串类型。这一点对Objective-C的新手来说非常容易令人困惑。所以，简单点说，只需要记住在Objective-C里面用@开始才能生成字符串。

如果你忘了加@，Xcode就会提示出错，具体的错误信息跟你使用的Xcode版本和语句本身都有关系。

把@放在字符串里是一个错误的做法，比如“@this is wrong”就没法通过编译。而且你必须用双引号，单引号也不行’this will not work’。在Objective-C里面，单引号只用于定义单个字符，而不是字符串。而且还需要注意的是，这个双引号必须是英文输入的，而不是中文输入法里面的双引号。

总结一下：

// 在Objective-C 中正确使用字符串的方法:
@"I am a good string"

// 下面都是错的:
"I forgot my @ sign"

"@My at-sign is in the wrong place"
'I should have double quotes'
@“My quotes are too fancy”

在NSlog这行代码里面，用到的字符串是@"The value of the slider is now: %f" 可能你不明白%f是干吗用的。这个就是传说中的“格式说明符“。你可以把它当做一个占位符：@"The value of the slider is now: X"，这里的X将被滑动条的数值所替代。在Objective-C中（其它编程语言也差不多类似），经常用这种方式来生成一个格式化字符串，然后在逗号后面用变量的数值来指定占位处的实际数值。

让变量来帮忙

在调试面板中用NSLog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，Objective-C（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CrazyDragViewController.m，然后修改@interface CrazyDragViewController ()这行代码为：

```
@interface CrazyDragViewController (){

    int currentValue;

}
```

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{}花括号里面，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。

还记得帮主当年的话吗？由内到位都要美。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。

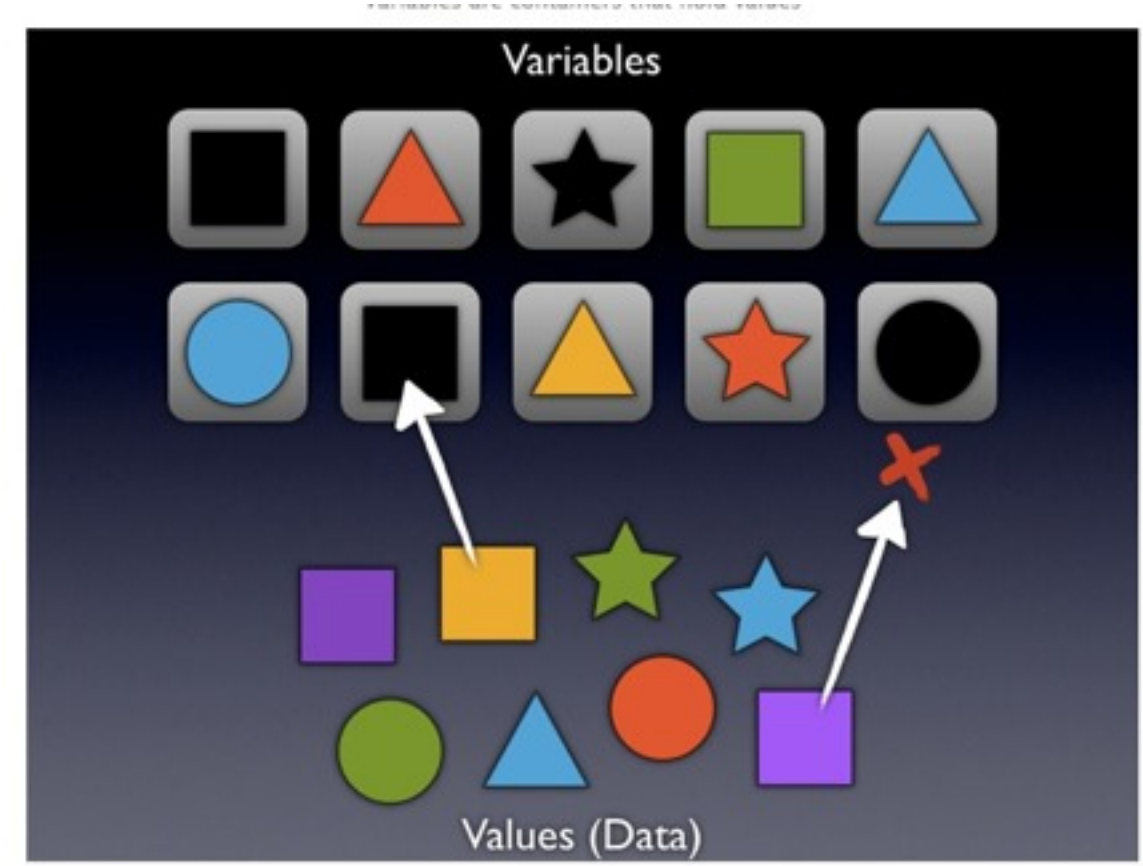
在之前的教程中曾提过视图控制器，或是任何一个对象都有自己的数据和功能。showAlert和sliderMoved动作就是功能的典型例子，而currentValue变量则是数据的一部分。

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。这就是变量的本质-变。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，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CrazyDragViewController。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。

好了，今天的学习到此结束。练习，理论，代码，就这样。你满意了吗？

送福利一张,这次是两个mm ~