

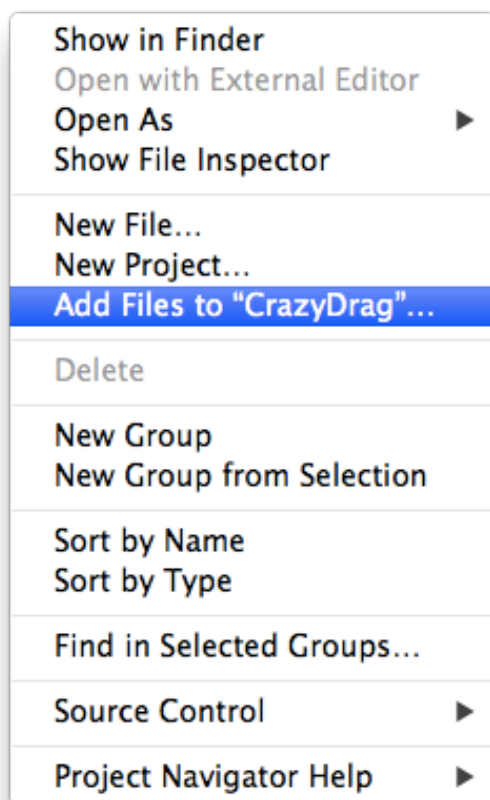
继续我们的iPhone开发学习。这里我们还将继续美化游戏的界面。

去掉状态栏只是万里长征第一步，我们还需要让界面变得更漂亮。

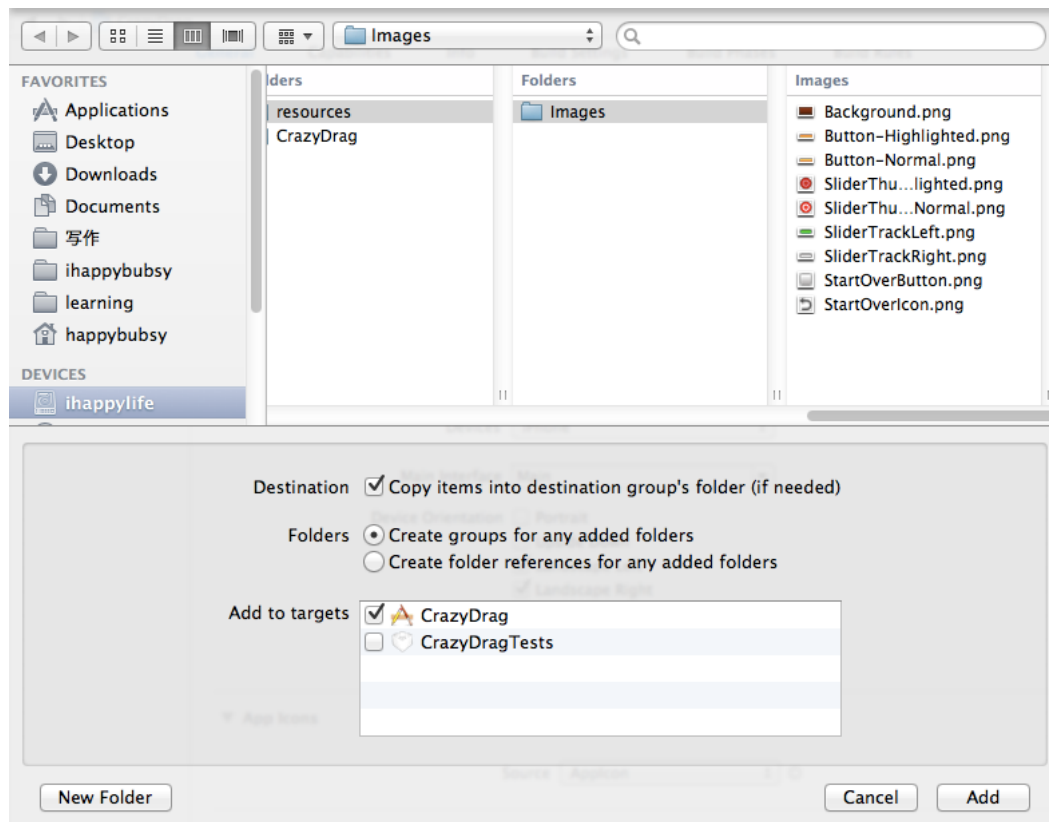
实际的控件和动作并没有发生变化，不过我们将要用图片来美化界面，同时调整一下一些颜色，字体设计（正是产品和设计人员的拿手好戏）。

需要注意的是，在iOS产品开发中，我们应尽可能使用PNG格式的图片。你可以完全使用自己的图片，我这里也准备了几个备用的。

在Xcode中右键单击Project Navigator（项目导航）下面的项目名称，然后选中Add Files to “CrazyDrag”...



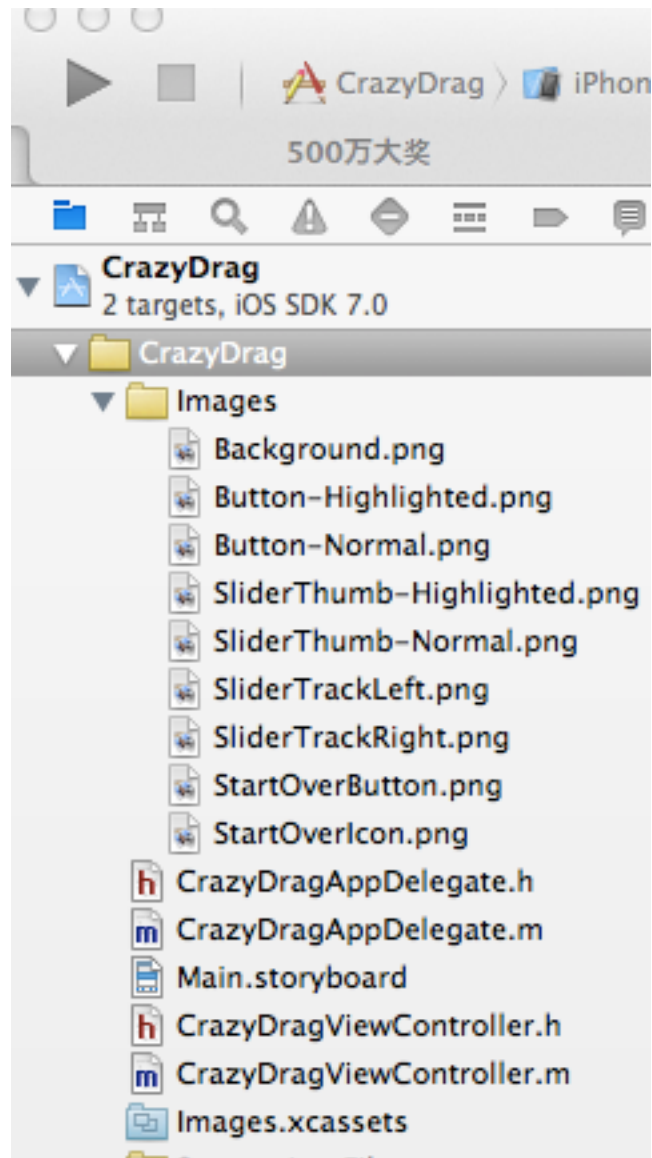
接下来找到Images这个子目录，



记得一定要选中“Copy items into destination's group folder (if needed)”和“Create groups for any added folders”。！！！！！！

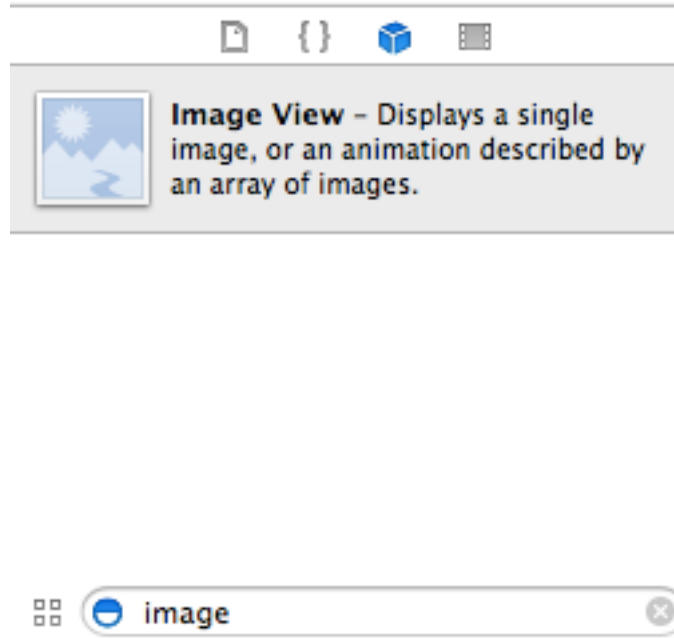
这一点一定要切记，否则你的游戏很可能在后面崩溃而你完全不知道为什么。

现在在Project Navigator（项目导航）下面就有了一个新的Images的群：



准备就绪了，让我们首先更换背景图片吧。

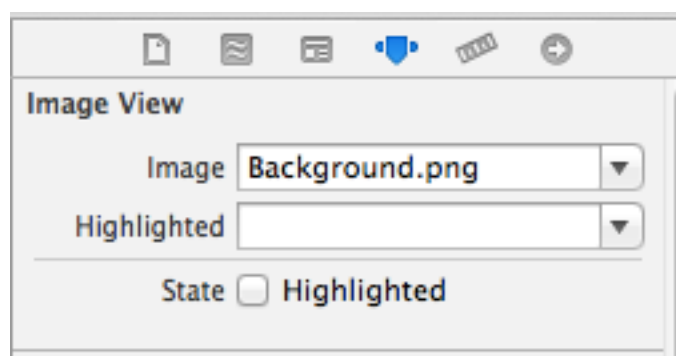
在Xcode中打开Main.storyboard，在Xcode右侧面板的Object Library中找到Image View。



把这个Image View拖到我们已有的用户界面上，先别管放在哪儿，待会儿我们会调整位置。

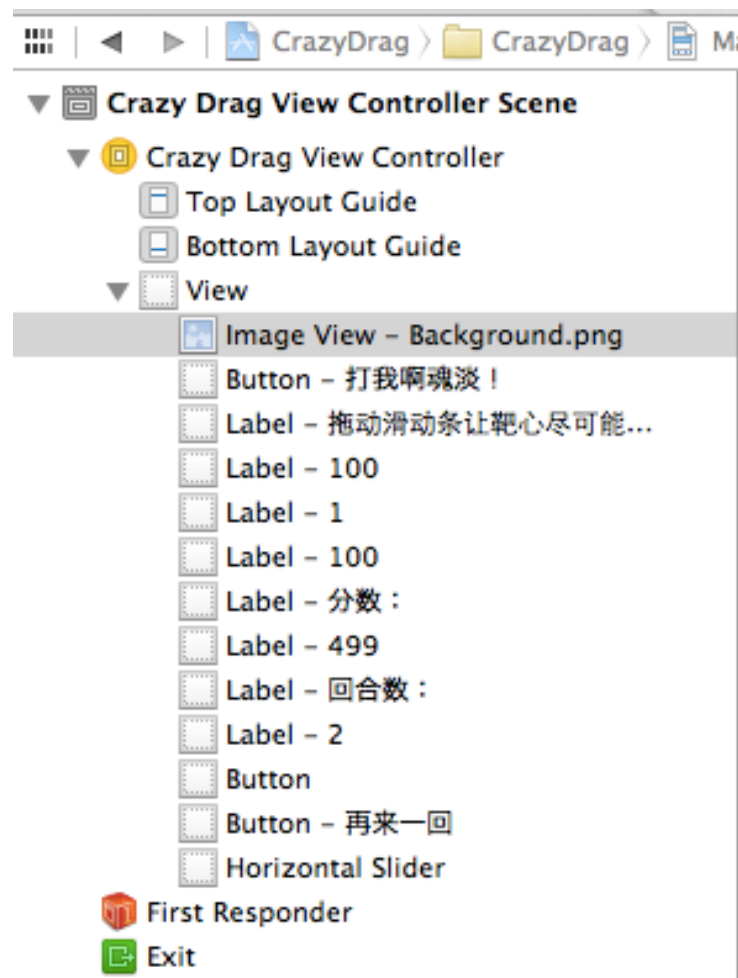
选中Image View，在右侧的面板中切换到Size Inspector，，设置X和Y为0，Width为568，Height为320。这样图片就会覆盖整个屏幕。

然后继续选中Image View，在右侧的面板中切换到Attributes Inspector，按照下图的提示将Image部分选择为Background.png。



当然，现在背景图片覆盖了其它控件。我们需要让其它控件的显示在背景之上。这一点很简单。

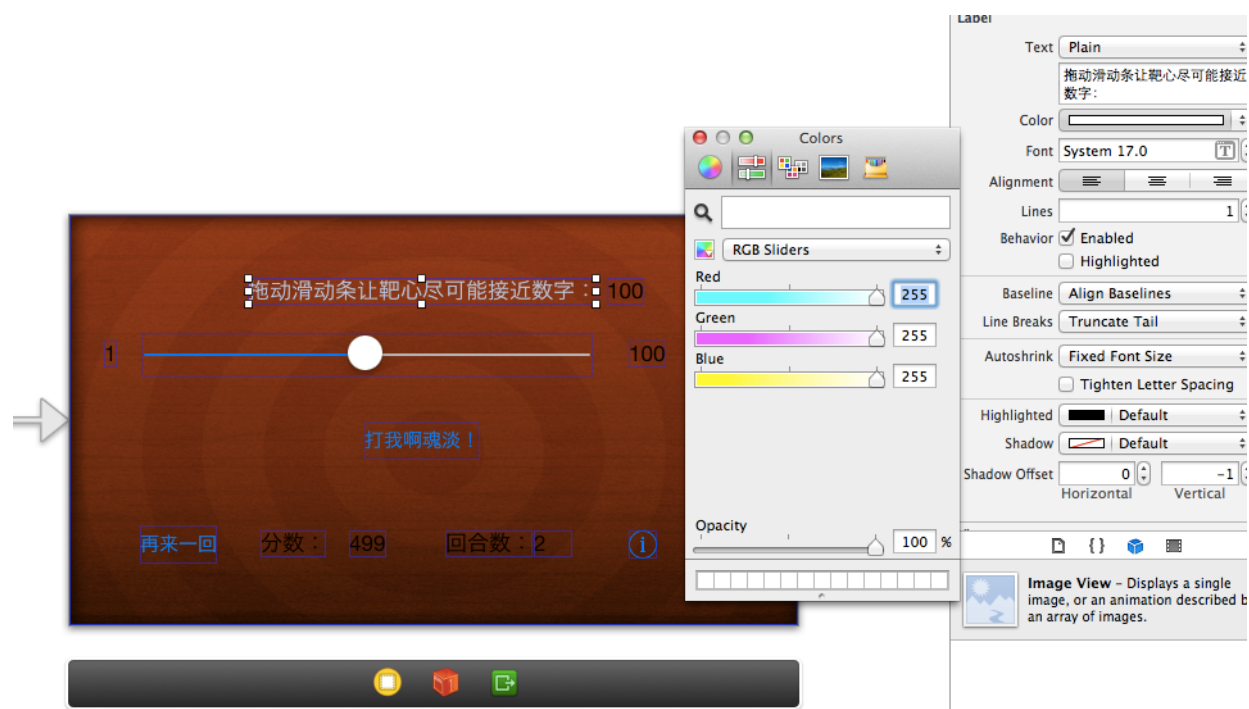
在Xcode的菜单栏中选中Editor,选择Arrangement- Send to Back，就好了。或者有一种更直观的方式（我个人倾向使用），在Interface Builder的Objects 面板中把Image View拖到最上面，就可以了。



一切就绪后运行下看看效果怎样。  
让我们继续前进，接下来我们将优化标签的显示。

因为背景图片比较暗，所以需要让标签中文本的色彩变亮。

选择顶部的标签，在Xcode的右侧面板中切换到Attributes Inspector，然后点击Color（注意具体的位置跟Xcode的版本有关。自从Scott Forstall被赶出苹果后，Xcode在这些细节上是一塌糊涂），

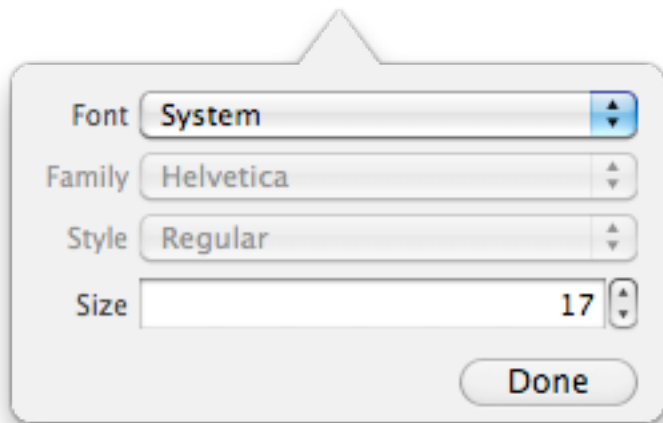


在色彩拾取器里面选择RGB Sliders，然后选择R:255,G:255,B:255,透明度100%，也就是纯白色，会用PS的都知道。

然后点击Shadow，将其设置为R:0,G:0,B:0,透明度50%。

然后把Shadow Offset设置为Horizontal:0,Vertical:1，这样阴影会显示在标签下面。

接着点击Font属性的[T]小图标，这里就可以设置字体了。你可以设置自己喜欢的字体。



你可以选择自己喜欢的字体，这里就不废话了。

然后把Autoshrink改为Fixed font size。

对于其它标签哥就不废话了，不然就没完没了了。具体怎么设置您说了算。

科普：关于iOS开发中所使用的字体

Interface Builder中所显示的字体是你的Mac电脑上有的字体，但不能确保在iPhone上也有。实际上iPhone所支持的字体要远远少于Mac上的字体。

怎么办呢？建议你到这个网站来看看iOS中可以用的字体：

<http://iosfonts.com>

接下来我们用类似的方式来美化按钮。我们可以给按钮添加背景图片，设置字体等等。具体如何操作就不再赘述了，最终的效果可能是类似下面的：



## 美化滑动条

对滑动条的美化要稍微复杂一些，事实上如果单纯使用Interface Builder的话我们只能稍微修改滑动条的外观。

这个时候，手写控件的灵活性就会体现出来了。

比如，为了设置按钮的色彩，我们既可以在Interface Builder中直接设置，也可以使用代码 setTitleColor:forState:来实现。当然，通过可视化的设计方式（所见即所得）可以大大提高开发效率。不过对于某些特殊的情况，比如这里的滑动条，我们也不得不直接手动控件来修改其属性。谁让苹果开发团队偷懒没给它提供在Interface Builder中直接修改的方式呢。

在Xcode中切换到CrazyDragViewController.m，在viewDidLoad方法中添加几行代码：



```

- (void)viewDidLoad
{
    [super viewDidLoad];

    UIImage *thumbImageNormal = [UIImage imageNamed:@"SliderThumb-Normal"];
    [self.slider setThumbImage:thumbImageNormal forState:UIControlStateNormal];

    UIImage *thumbImageHighlighted = [UIImage imageNamed:@"SliderThumb-
    Highlighted"];
    [self.slider setThumbImage:thumbImageHighlighted forState:UIControlStateHighlighted];



    UIImage *trackLeftImage = [[UIImage imageNamed:@"SliderTrackLeft"]
    stretchableImageWithLeftCapWidth:14 topCapHeight:0];
    [self.slider setMinimumTrackImage:trackLeftImage forState:UIControlStateNormal];

    UIImage *trackRightImage = [[UIImage imageNamed:@"SliderTrackRight"]
    stretchableImageWithLeftCapWidth:14 topCapHeight:0];
    [self.slider setMaximumTrackImage:trackRightImage forState:UIControlStateNormal];

    [self startNewGame];
    [self updateLabels];
}

```

在上面的代码中，我们为滑动条准备了四种图片：两个结点图片，两个滑动背景图片。结点图片和按钮类似，因此有一个正常状态，还有一个高亮状态。同时滑动条对于结点两边的滑动背景也采用不同的图片。左边的是绿色，右边的是灰色。

如果你看不懂上面的代码怎么办？老办法，按住option键，点自己需要看的方法,或者,就可以查看详细的帮助说明。

这里来看看对UIImage的使用吧。

```

UIImage *thumbImageNormal = [UIImage imageNamed:@"SliderThumb-Normal"];

```

在上面的这行代码中，我们创建了一个UIImage类型的变量，注意到我们在文件名的后面没有添加.PNG后缀。在iOS4之前是需要添加的，但之后都无需添加。

点击Run运行游戏，现在我们的游戏界面看起来就比较顺眼了！



继续前进-继续优化关于界面

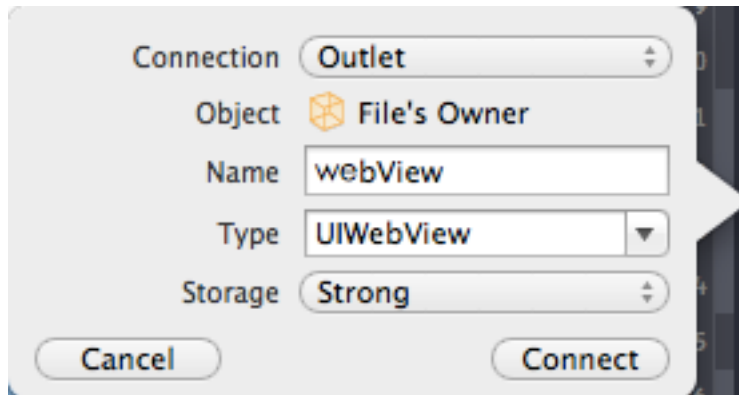
关于界面的内容还可以继续优化。

现在我们可以尝试用web view（网页视图）来替代text view（文本视图）。在Xcode中切换到AboutViewController.xib，然后选中文本视图，用delete键删掉它。然后拖一个Web view到刚才的位置，并让它占据刚才Text view所填充的空间。

web view（网页视图）顾名思义，可以用来显示网页。我们要做的就是给它一个到某个具体网站的URL网址。当然，这里我们会先让它来显示一个准备好的HTML页面。这样就无需联机下载才能看到内容。

在Xcode左侧的Project Navigator（项目导航）点击项目名称，然后选中Add Files，把我们准备好的CrazyDrag.html文件添加到项目中。这是一个HTML5文档，当然你也完全可以创建自己的HTML5文档添加进去。

接下来是添加属性变量。切换到AboutViewController.xib，选中Web View这个控件，然后按住control键，拖出一条线到Assistant Editor的@interface 和@end之间，按照下图来设置内容，将Name处改为webView，然后点击Connect创建关联。



最后当然是切换到AboutViewController.m，然后在@implementation AboutViewController这行代码之后添加一行代码：  
@synthesize webView;

最后在AboutViewController.m中找到viewDidLoad方法，更改其中的代码：

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view from its nib.

    NSString *htmlFile = [[NSBundle mainBundle]pathForResource:@"CrazyDrag"
ofType:@"html"];
    NSData *htmlData = [NSData dataWithContentsOfFile:htmlFile];
    NSURL *baseURL = [NSURL fileURLWithPath:[NSBundle mainBundle]bundlePath]];
    [self.webView loadData:htmlData MIMEType:@"text/html" textEncodingName:@"UTF-8"
baseURL:baseURL];
}
```

这一堆代码看起来又很恐怖，不过还是那句话，别恐慌。Don't panic!  
我们先大概解释下它的作用。它的主要作用是把本地的HTML文件加载到网页视图控件

中。首先我们在应用束中找到CrazyDrag.html文件，然后我们把它加载到一个NSData对象中，最后我们让网页视图显示NSData对象中的具体内容。

点击运行游戏，然后触碰(i)按钮，就会看到类似下面的界面。



以下是可以跳过的科普内容，如果你想对iOS开发进行更深入的了解，可以仔细看看，不然可以先暂时略过。

科普：NSBundle,NSData,NSURL

在刚才的这段代码中我们用到了三个新的类，分别是NSBundle,NSData,NSURL。这里简单介绍下这三个新的类。首先是NSBundle，它代表当前应用所在文件系统的位置，其中将代码和应用中所用到的资源组织在一起。NSBundle对象可以定位程序中的资源，动态加载可执行代码，同时可以协助进行软件本地化。mainBundle是NSBundle累的一个方法，它可以返回根当前应用所在的文件系统地址相关的NSBundle对象。pathForResource方法则是NSBundle对象的一个实例方法，它可以返回一个字符串，字符串的内容是某个特定名称和后缀文件的资源所在文件系统路径。该方法会首先在指定bundle的未本地化的资源目录中寻找匹配的资源文件（在iOS中通常是主bundle目录）。

如果没找到，则会继续在上一层的任一特定语言的'.lproj'目录中继续寻找。总之，在下面的这行代码中：

```
NSString *htmlFile = [[NSBundle mainBundle]pathForResource:@"CrazyDrag"
ofType:@"html"];
```

我们创建了一个htmlFile字符串变量，然后把CrazyDrag.html在当前应用文件系统中的路径保存在htmlFile变量里面。

接着看NSData这个类。NSData顾名思义就是和数据相关的类。这里的NS是前缀，在Objective-C中很多重要的类都是NS开头。NS其实是NeXT，也就是帮主当年被苹果驱逐期间所创办的一家电脑公司，这家公司以昂贵的NeXT硬件和优秀的NeXT操作系统而著称。最后苹果收购了NeXT，帮主回归，后来NeXT操作系统和Mac之前的版本融合在一起，最终形成了如今的Mac OS X操作系统。Data当然是数据的意思。NSData的作用是提供数据对象，以及数据缓冲区。dataWithContentsOfFile这个方法的作用是，根据某个指定的路径读取文件中的每个字节，然后创建并返回一个包含了其中所有内容的数据对象。

NSURL，顾名思义是NS和URL的组合，NS就不说了，看到它你就知道自己是在开发Mac或者iOS应用。URL也很明显就是网址的意思（Uniform Resource Locator）。不过这里我们用的不是类似http://之类的互联网地址，而是本地网址。fileURLWithPath这个方法的作用是根据指定的路径初始化并返回一个新的NSURL对象。

再来看最后一行代码：

```
[self.webView loadData:htmlData MIMEType:@"text/html" textEncodingName:@"UTF-8"
baseURL:baseURL];
```

在这行代码中，网页视图通过调用loadData:MIMEType:textEncodingName:baseURL:这个方法加载具体的网页内容。其中htmlData也即刚才生成的数据对象，MIMEType就是MIME类型，是指设定某种扩展名的文件使用何种浏览器插件打开，这里不多说textEncodingName是指字符的编码方式，比如UTF-8是指Unicode编码，可以显示中文，日文，英语，以及多种语言。与之对应的还有ASCII编码，它是专门针对英语设计的。看到这里，我们对刚才的这段代码应该有了更深的理解。

首先，我们创建了一个字符串对象，让它保存CrazyDrag.html在文件系统中的路径。

紧接着，我们创建了一个数据对象，让它保存CrazyDrag.html中的具体内容。

接着，我们创建了一个URL网址对象，让它保存系统的主路径。

最后，我们让网页视图以特定的形式来加载具体的网页内容。

好了，假如我们这里不想用固定的网页，而是想直接显示自己的个人网站内容怎么办？很简单，用下面的这段代码代替就好：

```
-(void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view from its nib.
```

```
// NSString *htmlFile = [[NSBundle mainBundle]pathForResource:@"CrazyDrag"
ofType:@"html"];
// NSData *htmlData = [NSData dataWithContentsOfFile:htmlFile];
// NSURL *baseURL = [NSURL fileURLWithPath:[NSBundle mainBundle]bundlePath]];
// [self.webView loadData:htmlData MIMEType:@"text/html"
textEncodingName:@"UTF-8" baseURL:baseURL];

NSURL *url = [NSURL URLWithString:@"http://www.apple.com"];
NSURLRequest *request = [NSURLRequest requestWithURL:url];
[self.webView loadRequest:request];

}
```

我们注释了之前使用本地文件加载网页视图的代码，而使用真实的网址来替代。你可以试着使用Option键来查看帮助文档，来仔细研究下这段代码的具体含义。如果看不懂也没关系。

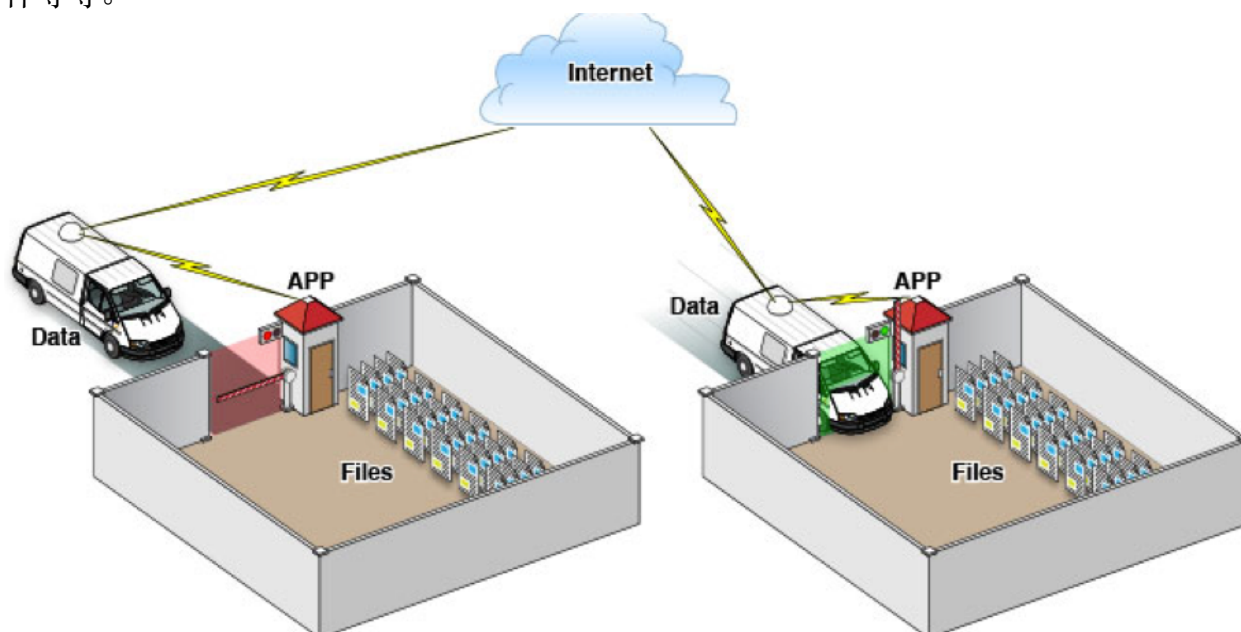
点击Run看看效果：



科普：关于iOS应用的沙盒机制和文件系统

具体还可以参考这里 (<http://bbs.weiphone.com/read-hm-tid-3903085.html>)

和我们熟悉的windows，android等应用的文件系统不同，所有的iOS应用只能在为该应用所创建的单独文件系统中读取文件，而不能访问其它应用的相关文件。人们通常称之为沙盒，几乎所有的非代码资源都会被放在这里，包括图形，图标，声音，属性列表，文本文件等等。



为了查看模拟器或设备的沙盒目录，我们需要在Mac系统下显示隐藏文件。

进入Terminal后输入`defaults write com.apple.finder AppleShowAllFiles -bool true`

重启finder后就可以在Library- Application Support- iPhone Simulator-x.x（模拟器的iOS版本编号）-Applications,然后会看到一长串的字母数字组合，进入后如果有个文件名称和项目名称一致，就是我们这个应用的沙盒目录了。

或者直接在Finder上点Go- Go to Folder，然后输入/Users/username/Library/Application Support/iPhone Simulator/

在沙盒中通常有3个文件夹，Documents, Library 和 tmp。因为应用的沙盒机制，应用只能在几个目录下读写文件

<Application\_Home>/AppName.app: 存放应用程序自身  
<Application\_Home>/Documents/: 存放用户文档和应用数据文件  
<Application\_Home>/Library/: 应用程序规范的顶级目录，下面有一些规范定义的的子目录，当然也可以自定义子目录，用于存放应用的文件，但是不宜存放用户数据文件  
<Application\_Home>/Library/Preferences，这里存放程序规范要求的首选项文件  
<Application\_Home>/Library/Caches，保存应用的持久化数据，用于应用升级或者应用关闭后的数据保存  
<Application\_Home>/tmp/，保存应用数据，但不需要持久化的，在应用关闭后，该目录下的数据将删除

iTunes在与iPhone同步时，备份所有的Documents和Library文件。  
iPhone在重启时，会丢弃所有的tmp文件。

具体请参考：

<http://developer.apple.com/library/ios/#documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/Introduction/Introduction.html>  
<http://marshal.easymorse.com/archives/3298>

好了，该休息一下了~





