# 让不懂编程的人爱上iPhone开发(2013秋iOS7版)-第15篇

不知不觉，我们已经在学习iOS开发的道路上前进了这么多。如果你仔细读过所有的内容，无论是唠叨，YY,实践，还是科普，那么你对iOS产品的开发应该有了一些基本的了解。
不过学无止境，而且在学习iOS开发的过程中，编码不是最重要的，设计和开发出满足用户需求的产品才是王道。永远记住这一点！无论是程序猿，产品经理或是设计师，都需要把这一点铭记在心。科技与艺术的结合才能打造真正优秀的产品，帮主早就为我们指明了前行的道路，至于如何做就看我们自己的了。

下面我们需要复习下图像分辨率的问题。

根据苹果最新发布的声明，从2013年5月1日起将不接受非Retina分辨率的应用，而且所有的应用必须针对iPhone5的1136*640分辨率进行优化。因此我们无需准备480*320的标清分辨率图片，而是一开始就完全针对Retina分辨率进行设计。

在iOS6中提供了强大的Auto Layout功能，如果我们采用的都是苹果标准控件，那么分辨率的问题将不再是问题。不过对于一些特殊的情况，比如游戏产品，以及背景图片，则需要做一些针对性的代码优化。比如首先判断设备的尺寸，然后根据不同的尺寸设置不同的背景图片。考虑到iOS应用的背景图片往往比较简单，通常可以用1136*640即可，因为Auto Layout功能会自动帮我们处理好缩放问题。对于比较复杂的图形，则可能需要进行一些代码级别的调整。
具体如何进行，请参考我的个人博客：http://blog.sina.com.cn/s/blog_4b55f68601018j0h.html

继续前进，看看如何添加酷炫的淡入淡出切换效果。

在结束这个系列的教程之前，我不得不提一下Core Animation。Core Animation是iOS的系统框架之一，使用它可以给应用添加各种酷炫的动画，而且只需要短短几行代码就可以实现。
还是那句话，仅仅让代码跑起来，让应用可以正常运行还不够，我们得从一些微妙的细节出发，让用户体会到使用我们产品或游戏的爽快感觉。

这里将要添加一个简单的淡入淡出效果，也就是当“重新来过”的按钮被触碰后，会

使用一个过渡效果让这个过程显得不是那么突兀。

在Xcode中 切换到CrazyDragViewController.m，更改startOver方法的代码如下：

```
- (IBAction)startOver:(id)sender {

    //添加过渡效果
    CATransition *transition = [CATransition animation];
    transition.type = kCATransitionFade;
    transition.duration = 3;
    transition.timingFunction  = [CAMediaTimingFunction
functionWithName:kCAMediaTimingFunctionEaseOut];


    [self startNewGame];
    [self updateLabels];

    [self.view.layer addAnimation:transition forKey:nil];
}
```

在上面的代码中，startNewGame和updateLabels这两个方法调用的代码之前就有，所以这里无需解释。我们需要看看新添加的代码。在深入学习这段代码之前，我们先了解它的作用。
简单的来看，我们创建了一个动画，它的类型是淡入淡出，时间长度是1秒。最终的效果是从当前屏幕上所显示的内容切换到更改后的内容（startNewGame和updateLabels所导致的变化，滑动条回到中央位置，标签的数值被更改）。

再次点击Run运行，可以看看效果。
如果你足够仔细，会发现一些微妙的视觉体验差异。

在结束教程之前，我们还要给这个小游戏加点新东西。
任何一款优秀的游戏都是一款艺术作品，它包含了精美的画面，炫酷的动画，动听的音效。。。且慢，我们这款游戏似乎是默片？你可以接受苍老师的视频教程里面竟然没有声音吗？显然不能，我们这个小游戏也一样，要加点好听的。

首先来个 背景音乐怎么样？最近天后离婚了，就拿《我也不想这样》当背景音乐吧。

在CrazyDragViewController.m的文件顶部添加一行代码：

#import <AVFoundation/AVFoundation.h>

这就引入了一个新的框架，所谓的AVFoundation框架，此AV非苍老师的AV，但这个AVFoundation框架的主要作用就是播放音频视频，所以苍老师也用得着。

接下来在@interface和@end之间的@property属性变量声明部分添加一行代码：

@property(nonatomic,strong)AVAudioPlayer *audioPlayer;

这就添加了一个AVAudioPlayer类型的播放器变量。

然后在@implementation之后添加一行代码，

@synthesize audioPlayer;

紧接着这行代码添加以下代码：

```
-(void)playBackgroundMusic{

   NSString *musicPath =[[NSBundle mainBundle]pathForResource:@"no"
ofType:@"mp3"];
   NSURL *url = [NSURL fileURLWithPath:musicPath];
   NSError *error;

   audioPlayer = [[AVAudioPlayer alloc]initWithContentsOfURL:url error:&error];
   audioPlayer.numberOfLoops = -1;
   if(audioPlayer ==nil){
      NSString *errorInfo = [NSString stringWithString:[error description]];
      NSLog(@"the error is:%@",errorInfo);
   }else{
      [audioPlayer play];
   }



}
```

这段代码的主要作用就是指定背景音乐所在的位置，然后创建并初始化一个播放器变量，设置循环播放。如果初始化失败，就提示出错的原因，如果成功，就直接播放背景音乐。

最后在viewDidLoad方法中添加一行代码：

```
   //播放背景音乐
   [self playBackgroundMusic];
```

我在resources文件夹里面放了天后的一首《我也不想这样》，你可以把这个no.mp3文件拖到项目中。
完成后点击Run，就可以在玩游戏的同时欣赏天后美妙动听的声音了。


小练习：
你可以从网上找一些或者自己制作一些短音效，然后在游戏交互的地方添加一些小音效。

科普：iOS常用框架-官方和第三方

iOS系统源自Mac系统，它的结构分为四个层次：


最下面的一层是核心操作系统层(Core OS Layer），这部分和Mac系统基本上完全相同，当然在电源管理等方面存在着一些细小的差别，后续的Mac系统改进也从iOS系统中吸取了一些不错的内容。它包括内存管理、文件系统、电源管理以及一些其他的操作系统任务。它可以直接和硬件设备进行交互。核心操作系统层包括以下组件：

OS X Kernel, Mach 3.0, BSD, SOCKETS, Power Mgmt, File System, Keychain, Certificates, Security, Bonjour

倒数第二层是核心服务层(Core Services)，同样和Mac桌面系统基本相同，但增加了移动设备的一些特性，比如Core Location和Map。主要可以通过它来访问iOS的一些服务：

Collections, Address Book,Networking, File Access,SQLite, Core Location, NET Services, Threading, Preferences, URL Utilities

第三层是媒体层(Media Layer)，和Mac系统基本相同，但有一些小的差异。通过Media Layer可以在应用中使用各种多媒体文件，录制音频和视频，绘制图形，或制作动画效果（比如我们这里用到的Core Animation)：

Core Audio, OpenGL ES, Audio Mixing, Audio Recording, Video Playback, JPG,PNG,TIFF,PDF,
Quartz, Core Animation

最上面的一层是界面交互层(Cocoa Touch),这一部分和Mac系统存在着非常大的差异，Mac的交互层称为Cocoa。它主要负责用户在iOS设备上的多点触摸交互操作，

包括以下组件：

UIKit（在Mac桌面系统对应的是AppKit）， Multi-Touch Events, Core Motion, Camera, View Hierarchy, Localization, Alerts, Web Views, Image Picker, Multi-Touch Controls

一般来说，开发iOS应用中一大半的时间都是在和Cocoa Touch这个层打交道，比如UIKit框架中的可视化组件，比如Image Picker获取照片库中的照片，通过Core Motion获取重力感应和陀螺仪的支持，以及获取用户通信录等等。

Framework（框架）-实质上就是帮我们实现各种特定功能的API类库。
在iOS产品开发的过程中，我们最经常接触到的就是Cocoa Touch，确切的说是UIKit Framework，所以任何一个iOS应用项目创建后都会自带UIKit Framework和Foundation Framework。当我们在开发产品的过程中需要用到一些特殊功能的时候，首先应该从最顶层寻找所需要的框架，只有顶层中的框架无法解决问题时，才能往下寻找相关的技术。
下面列出了一些常用的苹果官方提供的iOS 框架。

| 框架名称 | 功　能 |
| --- | --- |
| CoreLocation.framework | 使用 GPS 和 Wi-Fi 获取位置信息 |
| Foundation.framework | 提供 Object-C 的基础类（像 NSObject）、基本数据类型和操作系统服务等 |
| GameKit.framework | 为游戏提供网络功能：点对点互联和游戏中的语音交流 |
| MapKit.framework | 为应用程序提供内嵌地图的接口 |
| MediaPlayer.framework | 提供播放视频和音频的功能 |
| MessageUI.framework | 提供视图控制接口用以处理 E-mail 和短信 |
| OpenGLES.framework | 提供简洁而高效的绘制 2D 和 3D 图形的 OpenGL API 子集 |
| QuartzCore.framework | 提供动画特效以及通过硬件进行渲染的能力 |
| StoreKit.framework | 为应用程序提供在程序运行中消费的支持 |
| SystemConfiguration.framework | 检测当前网络是否可用和硬件设备状态的能力 |
| UIKit.framework | 创建和管理应用程序的用户界面 |

| 框架名称 | 功　能 |
| --- | --- |
| CoreLocation.framework | 使用 GPS 和 Wi-Fi 获取位置信息 |
| Foundation.framework | 提供 Object-C 的基础类（像 NSObject）、基本数据类型和操作系统服务等 |
| GameKit.framework | 为游戏提供网络功能：点对点互联和游戏中的语音交流 |
| MapKit.framework | 为应用程序提供内嵌地图的接口 |
| MediaPlayer.framework | 提供播放视频和音频的功能 |
| MessageUI.framework | 提供视图控制接口用以处理 E-mail 和短信 |
| OpenGLES.framework | 提供简洁而高效的绘制 2D 和 3D 图形的 OpenGL API 子集 |
| QuartzCore.framework | 提供动画特效以及通过硬件进行渲染的能力 |
| StoreKit.framework | 为应用程序提供在程序运行中消费的支持 |
| SystemConfiguration.framework | 检测当前网络是否可用和硬件设备状态的能力 |
| UIKit.framework | 创建和管理应用程序的用户界面 |

除了这些官方提供的框架，一些热心的程序猿（通常是老外，我朝程序猿热心开源项目的不多，开源中国上的还有些热血青年，其它的老油条都只顾自己赚钱去了）还在开发自己的项目之余搞了一些非常有用的iOS类库。

这里列出了raywenderlich网站所推荐的10款最有用的第三方iOS类库（框架）：

1.AFNetworking
轻量级而又超级高效的iOS网络编程框架，它支持iOS5.0及以上版本。
关于这个框架的使用，大家可以参考：
How To Create an App Like Instagram With a Web Service Backend – Part 1

2.SSToolKit
非常简单有用的工具，除了提供一些UI界面元素外，它还可以帮我们解决一些常见问题

3.GPUImage

用来处理图片的好框架，可以进行实时照片和视频处理，使用GPU而不是CPU，从而大大提高了程序运行效率。它比苹果官方的Core Image还要快，不过它缺少一些更先进的功能，比如Core Image的面部识别。

4.SocketRocket

如果你需要在iOS开发中和Web sockets打交道，显然不能错过这个框架。它可以轻松实现单一TCP连接的双工交流。虽然只有一些浏览器支持它，但对于一些实时在线应用来说非常有用。

5.HocketKit

可以使用它进行beta版发布，以及应用内崩溃报告，可以轻松发现程序中错误的来源。

6.JSONKit

超级NB的一个框架，可以高速解析JSON。当然从iOS5之后苹果官方也提供了自己的原生JSON支持（NSJSONSerialization)

7.MagicalRecord

Core Data是iOS开发中一个令人头疼的问题，不过MagicalRecord可以帮你减轻这方面的烦恼。

8.RestKit

可以让你的应用轻松的和REST远程API整合在一起。它可以处理网络事务，解析XML或JSON，并转换成你自定义的类。如果对它感兴趣，可以看看这个教程：Introduction to RESTKit Tutorial

9.Test Flight

可以用它来实现在线beta版测试的云服务，

10.MBProgressHUD

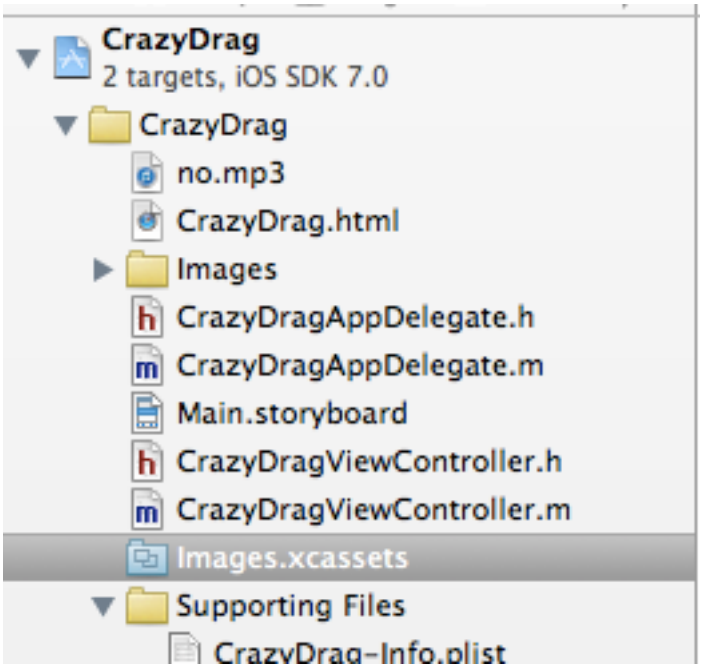
向用户提供进度条的视觉反馈效果。

或许上面这些东西目前对你来说没有什么意义，不过终有一天你会对这些东西感兴趣的相信我！

好了，让我们接着了解一个新东西-图标

其实这个东西一点都不新，只要你通过手机上或itunes上的app store下载过东西，就会明白对于一款iOS产品来说，图标是多么的重要！

在Xcode中点击左侧导航栏的Images.xcassets

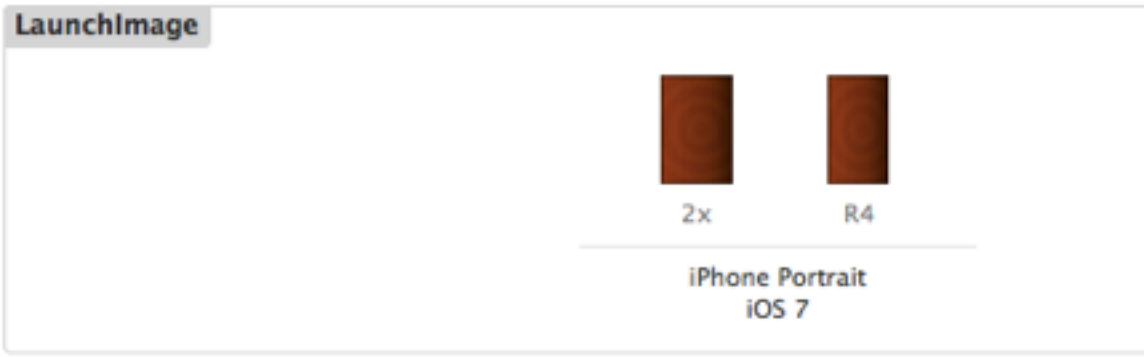



首先来添加AppIcon

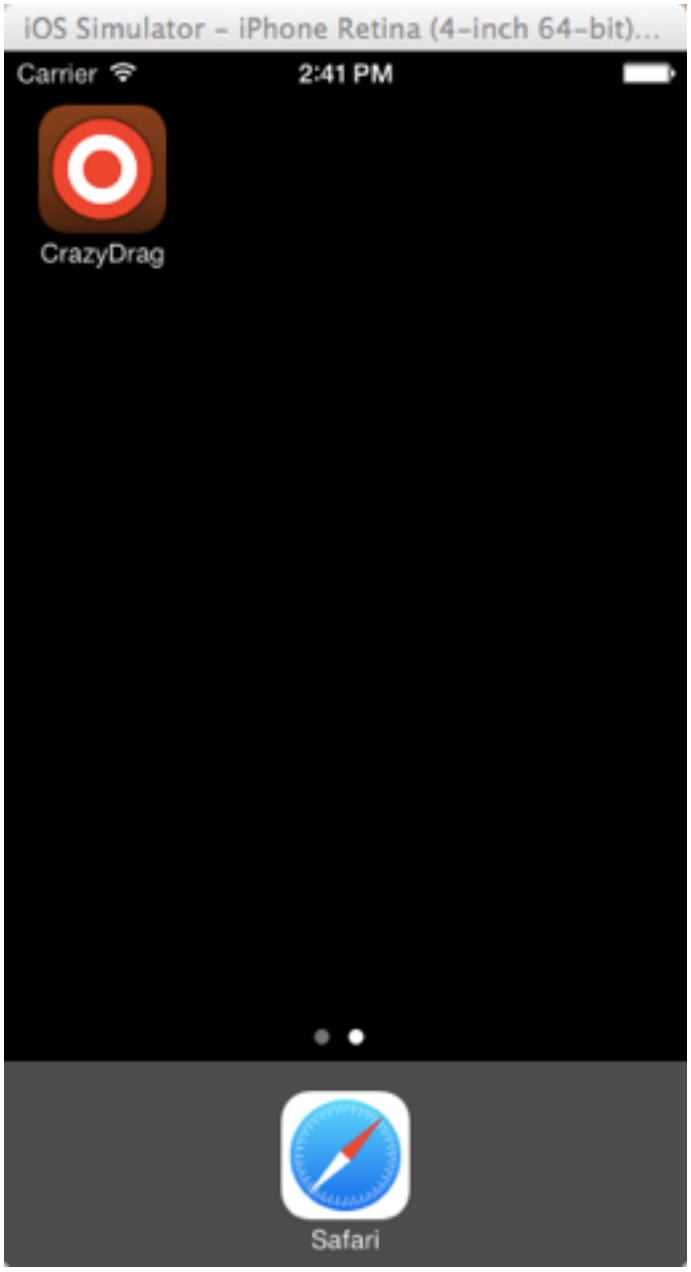

注意这里需要58*58像素，80*80像素和120*120像素的图片。

直接从Resources的Icon文件夹中把对应的文件拖到相应的位置就好了。放好后的显示如下：


除了图标之外，我们还得准备启动画面所需要的图片。如果不提供这种图片，产品在启动的时候就会显示黑屏，知道加载主界面成功。
很多开发者选择类似电影开场的过场动画，或者是公司logo，其实这是苹果强烈反对的做法。在官方的HIG用户界面指南中，建议使用和首界面相似的静态图片。
切换到LaunchImage，添加两张图片，分别是640*960（2X那里）和640*1136(R4那里）。完后后的效果如下：




编译运行项目，你就会发现模拟器中的应用图标已经更改了。




补充一点，对于我们这款应用，这几个图片已经够用了，但是ipad游戏会怎样？苹果对于iOS7应用/游戏的图标和启动画面要求如下：

**Table 38-1** Size (in pixels) of custom icons and images

| Description | Size for iPhone 5 and iPod touch (high resolution) | Size for iPhone and iPod touch (high resolution) | Size for iPad (high resolution) | Size for iPad 2 and iPad mini (standard resolution) |
|---|---|---|---|---|
| App icon (**required** for all apps) | 120 x 120 | 120 x 120 | 152 x 152 | 76 x 76 |
| App icon for the App Store (**required** for all apps) | 1024 x 1024 | 1024 x 1024 | 1024 x 1024 | 1024 x 1024 |
| Launch image (**required** for all apps) | 640 x 1136 | 640 x 960 | 1536 x 2048 (portrait) 2048 x 1536 (landscape) | 768 x 1024 (portrait) 1024 x 768 (landscape) |
| Spotlight search results icon (recommended) | 80 x 80 | 80 x 80 | 80 x 80 | 40 x 40 |
| Settings icon (recommended) | 58 x 58 | 58 x 58 | 58 x 58 | 29 x 29 |
| Toolbar and navigation bar icon (optional) | About 44 x 44 | About 44 x 44 | About 44 x 44 | About 22 x 22 |
| Tab bar icon (optional) | About 50 x 50 (maximum: 96 x 64) | About 50 x 50 (maximum: 96 x 64) | About 50 x 50 (maximum: 96 x 64) | About 25 x 25 (maximum: 48 x 32) |
| Default Newsstand cover icon for the App Store (**required** for Newsstand apps) | At least 512 pixels on the longest edge | At least 512 pixels on the longest edge | At least 512 pixels on the longest edge | At least 256 pixels on the longest edge |
| Web clip icon (recommended for web apps and websites) | 120 x 120 | 120 x 120 | 152 x 152 | 76 x |

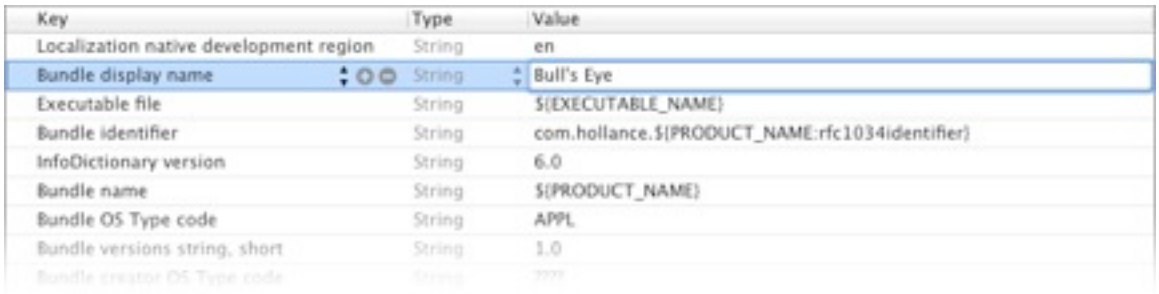


图标有了，启动画面有了，我们还需要有一个好的产品名称。

之前我们用了简单的名称CrazyDrag，但我希望在Appstore里面显示的是天天拖飞机，或者天天土豪金，总之和人类的表达习惯更接近。不过需要注意的是，在图标下面的空间非常有限，你的产品名称也不能太长了，否则就无法显示完全。

再一次，改变CrazyDrag-Info.plist。

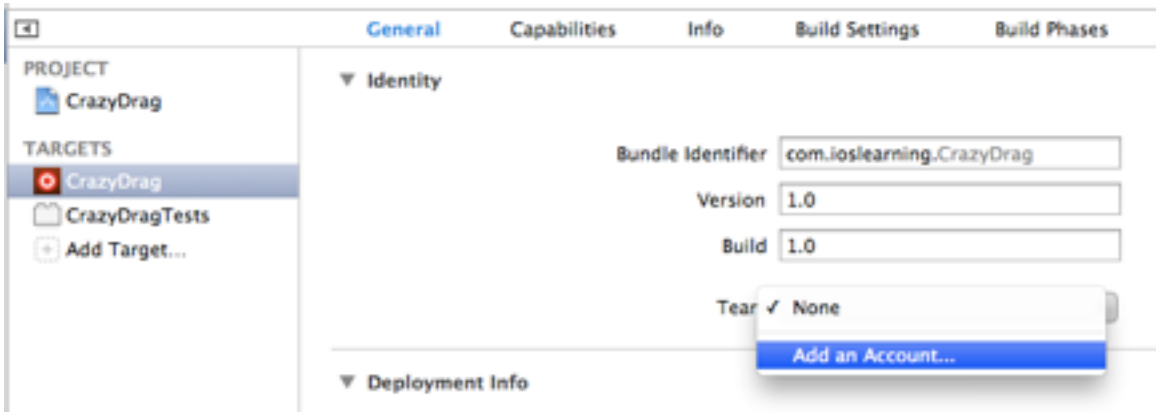
我们会看到在key这一列有一个键的名称是Bundle display name，当前的内容是${PRODUCT_NAME}。这就意味着Xcode会自动把项目名称CrazyDrag作为产品名称。这里我们可以替换成自己想要的任何名称。



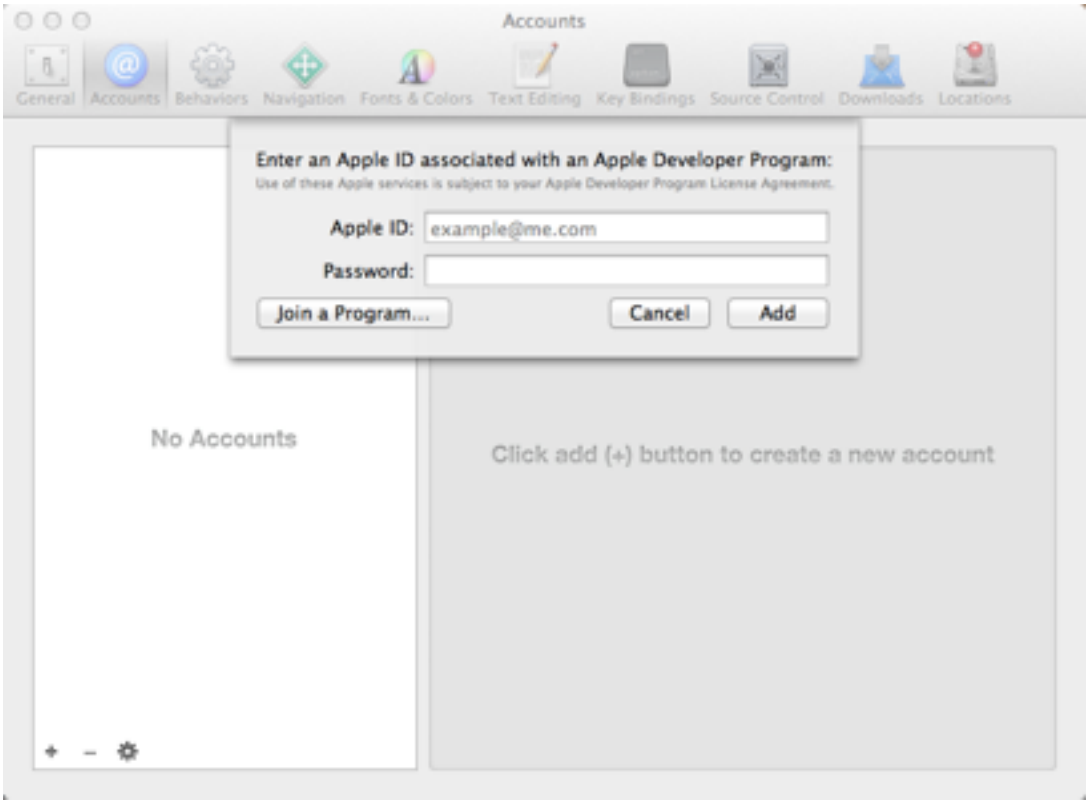

好了，现在点击Run来运行，就可以在模拟器上看到图标和更改后的产品名称了，打开应用后还会看到启动画面。

最后的最后，让我们在设备上实际测试这款应用，这部分就不废话了，请参考：http://www.cocoachina.com/bbs/read.php?tid=2776

不过在Xcode5中有更简单的方法，假定你已经注册了苹果的开发者账号。那么可以在Xcode中点击TARGETS下的CrazyDrag，切换到General选项卡，然后在Identity下面有一个Team,最开始当然是没有东西的。



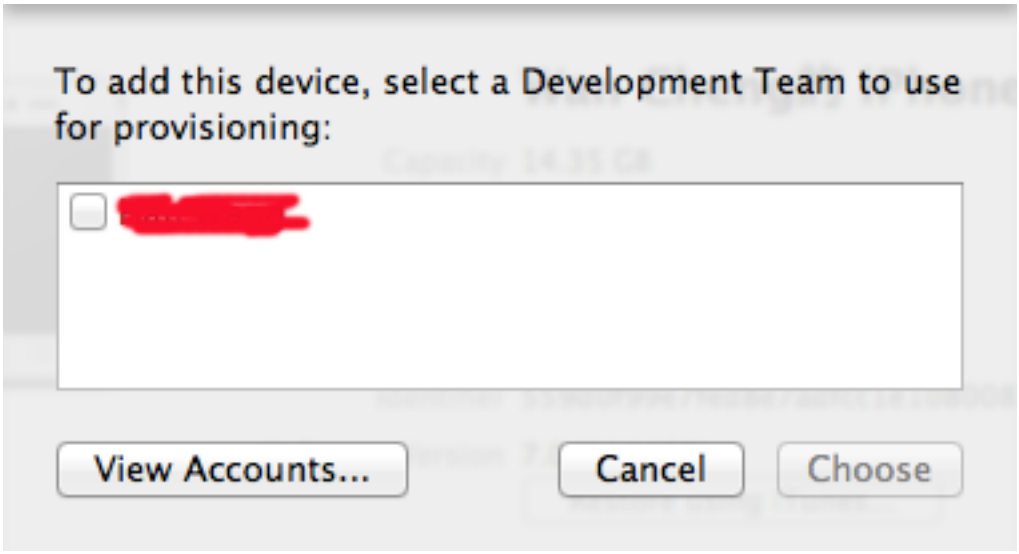

点击Add an Accout:




输入自己的开发者账号和密码，点击Add即可。

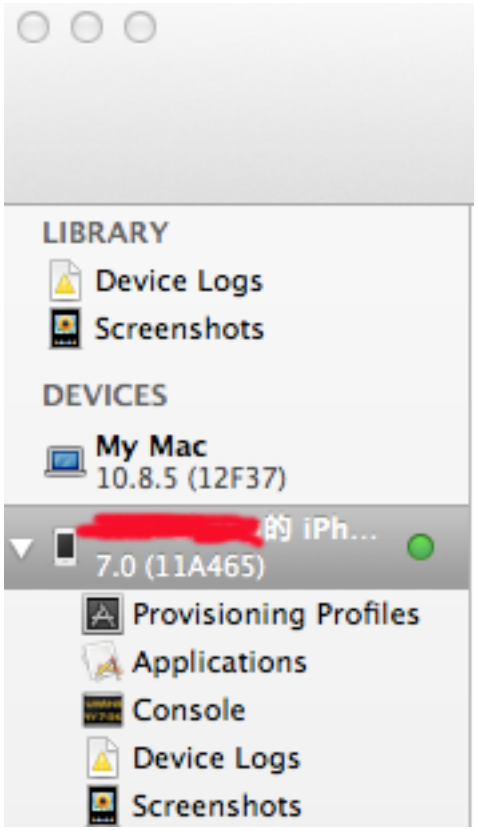
然后把自己的设备连接到Mac电脑上，点击Xcode工具栏上的Window-->Organizer，看到下面的画面：


我用红色圈出了一个按钮，点击这个按钮Use for Development，会弹出下面的对话框：




勾选红色涂抹（一个账号），然后点击Choose，就好了。

你会看到左侧的设备列表旁边出现绿色的圆形，表示该设备可以用作测试设备了。不过前提是，你的设备也是最新的ios7哦。




回到Xcode主界面：

选择刚才的设备，然后点击Run，就可以在设备上看到这个游戏了，是不是很有成就感呢？


当然，这是在iphone4s上跑出来的效果。

好了，到了这里，我们的系列教程第一部分已经结束。

希望在我们一起学习iPhone开发的这个过程中，对于iOS产品的开发可以形成一个整体的认识。
在后续的系列教程中，我们也期待着更多收获。如果你懒得看后续的教程，也可以到这里就结束，然后自己去探索iOS产品开发的更多未知秘密。祝你好运！

最后的福利：


昵图网 nipic.com/天堂树

博客：blog.sina.com.cn/eseedo

微博：eseedo

微信：iseedo

Email: eseedo@gmail.com