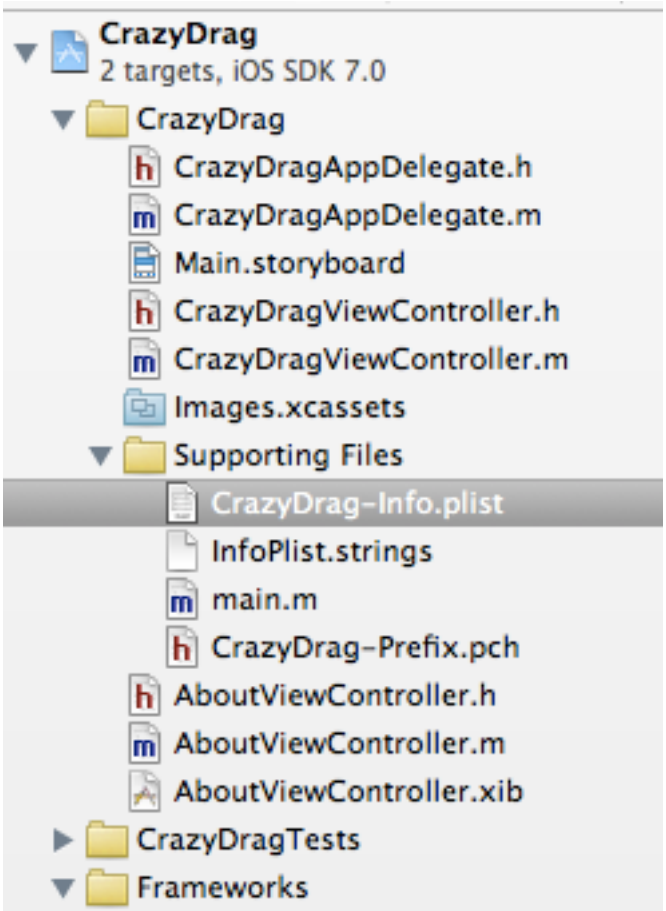
让不懂编程的人爱上iPhone开发(2013秋iOS7版)-第13篇

欢迎继续我们的iPhone开发学习。
在上一篇的内容中，我们学会了如何从单界面升级为多界面应用，还接触了一些看似比较高深难懂的概念。
为了补偿大家，这一篇的内容应该是产品和设计人员的最爱-美化界面。
为了让玩家得到更好的沉浸体验，我们必须让产品的外在和内在一样美，甚至更美。

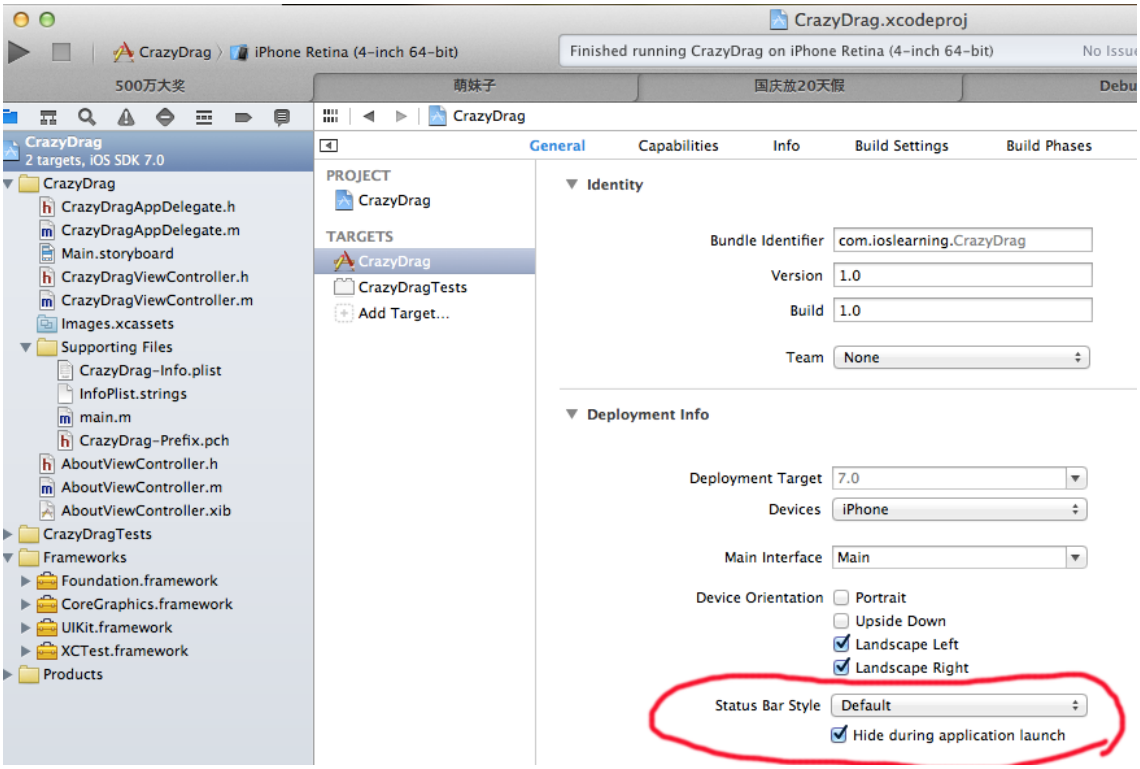
有一个小瑕疵要调整下。如果你注意看about这个界面，会发现上面的状态栏（Carrier，电池神马的）看起来很恼火，非常的不和谐。那么怎么去掉呢？很简单。
在Xcode中，在项目导航（左侧）找到Supporting Files,里面有个CrazyDrag-Info.plist文件，点击在右侧查看其内容。



其中有很多信息，具体的我们会在后面进行详细说明，现在只需要知道它的作用是配置一

些项目的基本信息，它的本质是一个xml文档。

右键点击下面的空白区，选择Add Row,在Key那一栏选择“View controller-based status bar appearance”，然后在Value那里设置为NO。
这还不够，在Xcode中点击项目名，选择TARGETS下面的CrazyDrag，然后在右侧的Deployment Info下面，选中Status Bar Style下面的Hide during application launch，如下图。



好了，搞定了。
点击Run跑一下，会看到游戏界面和关于界面的状态栏都消失了。

接下来是两个科普，如果你是新手，很可能会看不懂。你可以选择现在来攻坚，也可以选择等以后熟悉iOS开发了再深入去了解。

科普：关于Info.plist文件，XML

在任何一个iOS应用或游戏中，如同AppDelegate类一样，都有一个以项目名称开头的Info.plist文件，比如这里的CrazyDrag-Info.plist文件。它的内容通常由三列组成，最左边是Information Property List（属性列表），中间是Type（属性值的类型），而最右边则是Value（属性值）。
Info.plist文件其实是一个XML文档。XML其实就是可扩展性标记语言(extensible markup language)，它并非iOS中所特有的，在几乎任何一种编程语言的使用过程中，我们都会碰到XML文档。XML是所谓标准通用标记语言(SGML）的子集，其作用是以规范的形式（成对出现的标记）来保存数据。XML与传统的Access,Oracle,SQL Server, MySQL数据库不同。传统的数据库功能强大，提供了强大的数据存储和分析能力，而XML仅仅用来存储数据，需要自行编写代码来进行数据的分析和处理。但XML的好处是它超级简单易用，可以在任何语言编写的任何应用程序中读写数据，已经成了网络数据交互的唯一公共语言。
你可能不知道XML文档，但或许多半听说过HTML文档吧。XML文档只不过是HTML文档的规范式表达。它们的区别在于，XML的核心是数据内容本身，而HTML的核心是如何显示数据。
plist文件的本质就是XML文档，只不过其中的内容都和iOS应用的相关设置有关。在Xcode中右键单击CrazyDrag-Info.plist,选中open as,选中source code，就可以看到下面的内容：

<?xml version="1.0" encoding="UTF-8"?>

```
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>CFBundleDevelopmentRegion</key>
    <string>en</string>
    <key>CFBundleDisplayName</key>
    <string>${PRODUCT_NAME}</string>
    <key>CFBundleExecutable</key>
    <string>${EXECUTABLE_NAME}</string>
    <key>CFBundleIdentifier</key>
    <string>com.ioslearning.${PRODUCT_NAME:rfc1034identifier}</string>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>6.0</string>
    <key>CFBundleName</key>
    <string>${PRODUCT_NAME}</string>
    <key>CFBundlePackageType</key>
    <string>APPL</string>
    <key>CFBundleShortVersionString</key>
    <string>1.0</string>
    <key>CFBundleSignature</key>
    <string>????</string>
    <key>CFBundleVersion</key>
    <string>1.0</string>
    <key>LSRequiresIPhoneOS</key>
    <true/>
    <key>UIMainStoryboardFile</key>
    <string>Main</string>
    <key>UIRequiredDeviceCapabilities</key>
    <array>
        <string>armv7</string>
    </array>
    <key>UIStatusBarHidden</key>
    <true/>
    <key>UISupportedInterfaceOrientations</key>
    <array>
        <string>UIInterfaceOrientationLandscapeLeft</string>
        <string>UIInterfaceOrientationLandscapeRight</string>
    </array>
    <key>UIViewControllerBasedStatusBarAppearance</key>
    <false/>
</dict>
</plist>
```

很显然，plist文档其实就是满足苹果DTD标准的XML文档。

那么，Info.plist文件中这些键值的作用是什么呢？
这里大概说明了一下，更详细的可以参考苹果的官方文档，也可以参考（http://www.biancheng521.com/article/enauola/7853130.html）。
Localiztion native development region --- CFBundleDevelopmentRegion 本地化相关，如果用户所在地没有相应的语言资源，则用这个key的value来作为默认.
Bundle display name --- CFBundleDisplayName 设置程序安装后显示的名称。应用程序名称限制在10- 12个字符，如果超出，将被显示缩写名称。
Executaule file -- CFBundleExecutable 程序安装包的名称
Bundle identifier --- CFBundleIdentifier 该束的唯一标识字符串，该字符串的格式类似com.yourcompany.yourapp，如果使用模拟器跑你的应用，这个字段没有用处，如果你需要把你的应用部署到设备上，你必须生成一个证书，而在生成证书的时候，在apple的网站上需要增加相应的app IDs.这里有一个字段Bundle identifier，如果这个Bundle identifier是一个完整字符串，那么文件中的这个字段必须和后者完全相同，如果app IDs中的字段含有通配符*，那么文件中的字符串必须符合后者的描述。
InfoDictionary version --- CFBundleInfoDictionaryVersion Info.plist格式的版本信息
Bundle name ---CFBundleName产品名称
Bundle OS Type code -- CFBundlePackageType：用来标识束类型的四个字母长的代码,
Bundle versions string, short --- CFBundleShortVersionString 面向用户市场的束的版本字符串
Bundle creator OS Type code --- CFBundleSignature：用来标识创建者的四个字母长的代码
Bundle version --- CFBundleVersion 应用程序版本号，每次部署应用程序的一个新版本时，将会增加这个编号，在app store上用的。
Application require iPhone environment -- LSRequiresIPhoneOS:用于指示程序包是否只能运行在iPhone OS 系统上。Xcode自动加入这个键，并将它的值设置为true。您不应该改变这个键的值。
Main storybard file base name -- UIMainStoryboardFile 这是一个字符串，指定应用程序主nib文件的名称。
supported interface orientations -- UISupportedInterfaceOrientations 程序默认支持的方向。

科普：一个iOS应用究竟是如何启动的？

1.在任何C系语言（C,C++,Objective-C等）开发的应用中，程序的初始入口都是main.m。使用Objective-C开发的iOS应用当然也是如此。因此首先会从Supporting Files中的main.m开始。

#import <UIKit/UIKit.h>

```
#import "CrazyDragAppDelegate.h"

int main(int argc, char * argv[])
{
  @autoreleasepool {
    return UIApplicationMain(argc, argv, nil, NSStringFromClass([CrazyDragAppDelegate
class]));
  }
}
```

当然，这一切都是自动的。
2.接下来的事情也不用我们操心，系统会自动调用UIApplicationMain函数。

按住option键，点击UIApplicationMain，会看到关于它的帮助文档。它会做几件不同的事情，比如初始化一个应用对象，使用指定的类名称（这里的CrazyDragAppDelegate)初始化一个代理（如果有）。然后它会设置程序的主Event loop，也就是通用的无限循环。它还会读取Info.plist文件，如果其中有UIMainStoryboardFile键值，就读取它指向的storyboard文件来家族UIApplication对象的各种属性。如果没有storyboard（比如使用xib文档）其中有NSMainNibFile键值，就使用它所指向的nib文件(xib文件）来加载UIApplication对象的各种属性。
后面的不用多说了，连接都已经建立好，UIApplication对象就依次呼叫代理的不同方法，启动Event loop，那个通用的无限循环。
3.系统会自动调用CrazyDragAppDelegate代理类的几个方法，比如didFinishLaunchingWithOptions,applicationWillResignActive等等。
didFinishLaunchingWithOptions这个方法最为重要，这里创建并初始化了我们的主界面。
4.进入CrazyDragViewController这个主界面的视图控制器，然后等待和玩家之间的事件交互。
以上就是我们这款iOS游戏的启动过程。

当然，上面的两个科普内容对于一个新手来说有点太深入了。如果你觉得暂时看不懂，没有关系，可以先跳过。

如果你看完了，而且都理解了，那么恭喜你，你已经跨过了iOS开发的第一个门槛。

好了，又见福利时间。


PHOTO BY SUIYUAN
爱丽
www.aili.com