# 一、Sprite Kit 框架特性的简介

## 1.1 Sprite Kit 的特性简介

Sprite Kit框架是苹果在iOS7推出的一款用于2D游戏开发的框架，它面向iOS和OS X平台。

Sprite Kit的的底层渲染使用的是OpenGL的绘图命令，这些工作对于上层开发者不可见，所以这个框架对于游戏开发者来说，意义就在于可以让开发者投入更多的精力在游戏设计和业务逻辑的工作上面，而不是那些繁琐的技术细节。

这就是苹果推出Sprite Kit的框架所要达到的目的。

## 1.2 Sprite Kit 与其他框架的对比

Sprite Kit相对于其他开发框架主要有以下三个优势：

第一，它的架构设计十分的简洁，只有二十多个类就覆盖了所有的功能。

第二，可以与5以上版本的Xcode无缝对接，同时支持即将发布的Xcode6中新推出的Swift语言。

第三，它还有一个极大的亮点是内置了自己的物理引擎，可以非常方便的使用。

当然Sprite Kit本身也有它自己的局限性：

第一，Sprite Kit不能跨平台，它不支持Android平台的游戏开发。

第二，Sprite Kit由于推出的时间较晚，它的普及度不够高导致可以参考的资料和第三方工具都比较少。

# 二、第一个Sprite Kit项目

## 2.1 介绍Sprite Kit的框架结构

学习Sprite Kit框架最好的方法是在实践中观察，这个模块主要会讲解如何建立自己的第一个Sprite Kit工程，

首先我们先来了解一下整个游戏框架的结构，如果你之前对于Cocos2d有所了解的话，你就知道Cocos2d的整个游戏基本上是分为四层的，而Sprite Kit把它简化成了三层：

第一层是视图层，每个游戏都会有多个场景，所以在多个场景之间的调度切换需要有一个模块统一进行管理，而这一层的类模块主要实现的功能就是管理多个场景。这个类的名字叫SKView，SKView继承自UIView，它的主要功能是作为一个SKScene的容器来岁SKScene进行调度和管理。与Cocos2d中的CCDirector类的功能比较类似，然而Sprite Kit的SKView类更加简洁，它的实现遵从着苹果一贯的设计风格，去除了繁琐不重要的功能，只提供最核心的功能，所以非常容易学习使用。

第二层是场景层，场景也是一个容器，视图层负责的主要工作是管理不同的场景，而场景层负责的主要工作是画面的渲染优化以及处理游戏的内容与逻辑，在使用了物理引擎的项目中，场景层还负责进行物理模拟的计算。

第三层是内容层，Cocos2d在这一层之前还多了一个Layer图层用以判断图像渲染的层次，而这方面的工作在Sprite Kit框架中则使用了一个叫做NodeTree的机制完成了这个功能的实现。

对于Cocos2d来讲，CCLayer层是一个画布，而在Sprite Kit中，我们就可以直接把内容放入SKScene当中。

而这些内容类都继承自SKNode，其中最重要的类当然是SKSpriteNode，我们通过SKSpriteNode将游戏的角色实现后放入场景，并使用SKAction为角色添加动作然后我们就可以实现各种各样的需求，完成我们的开发了。

## 2.2 建立一个Sprite Kit工程

Sprite Kit 默认的模板使用了Storyboard，StoryBoard呢是苹果在xib之后推出的又一个拖拽流的工具，拖拽流的优点在于我可以简洁直观的创建自己想要的界面，缺点当然也很明显，在多人协作的场景下使用Storyboard会让项目变得难以维护，假如我们不想使用Storyboard这个工具的话我们应该怎么做呢？

第一步，先清除项目中的Storyboard文件

第二步，把plist文件中的Main Storyboard这个链接项去掉

第三步，补齐AppDelegate文件中的启动代码

第四步，在loadView方法中替换掉ViewController默认自带的UIView

然后我们就可以看到编译运行后的结果，

与不使用Storyboard时候的效果是一摸一样的

## 2.3 创建一个SKScene

虽然系统自带了一个写好的SKScene，我们还是重新新建一个SKScene来详细了解如何建立一个SKScene的过程，在前边的部分我们了解到了SKView是SKScene的容器，它负责管理调度SKScene，所以我们先来了解一下SKSView的配置。

首先是如何判断当前场景是否已经展现了SKScene，SKView自己有一个成员属性叫scene，这个属性将一直保持SKView当前展现的SKScene，假如这个成员变量为空的时候说明SKView当前没有展现SKScene。

接下来是SKView自备的一些方便的调试功能，

showFPS是用来显示帧速率的开关，帧速率是画面流畅度的最主要指标。

showNodeCount是显示当前场景内节点数量的开关，这个可以帮我们监控场景内容，帮我们及时释放掉不必要的节点来优化性能。

showDrawCount是显示当前场景内绘图次数的开关，通过这个参数我们可以监控性能，因为绘图次数过多必然会导致换面流畅度下降。

showPhysics与物理引擎的功能相关，它用来显示物理模型的外框。

设置好了SKView之后我们就来加入SKScene，

首先我们新建一个SKScene的class方法文件，并在ViewController中把它import进来，

SKScene的初始化方法一般主要是使用initWithSize方法，初始化之后呢我们还需要设置场景的缩放模式scaleMode，设置后好之后呢，我们就可以直接把他放入SKView当中了，

我们使用SKView中的presentScene方法就可以看到我们自己的SKScene的效果了。

## 2.4 给场景添加内容

接下来我们就在SKScene中添加自己想要的内容，在系统自带的样板里我们已经可以看到添加了文字的SKLabelNode，这次我们使用SK框架最主要的类SKSpriteNode来看看效果，首先我们要初始化这个类，我们就是用系统自带的图片，然后给这些类配置大小和位置属性。

编译运行我们发现了两个问题：

第一个问题是我们发现精灵的位置并不是我们想象的那样在左上方，这是因为坐标系统不一样

SKScene的坐标系统是跟我们正常的数学坐标系相同的方向的，blabla……

第二个问题是我们发现精灵的自身定位坐标并也不是从左上角计算的而是从中间开始的，

这是SK框架的Anchor系统，会默认从中点开始计算，这个会在后边的知识点中讲到。

然后我们来改变坐标了解一下坐标系统的工作原理。

首先是改变坐标：blabla……

其次是改变大小：blabla……

## 2.5 添加响应动作

SKScene继承自SKEffectNode，SKEffectNode又继承自SKNode，而SKNode则集成了一个UIResponder类，UIResponder类里边有方法可以响应我们的点击行为。

我们可以在SKScene中覆盖UIResponder的方法来实现对于屏幕点击事件的响应，通过方法传进来的touches参数中的信息我们就可以获取点击的详细信息，通过locationInNode方法我们可以获取location的坐标，并拿这个坐标来判断我们是否点击了某个精灵。

当然我们需要给精灵添加一个可以区分的标记，

最后我们来通过控制台输出的方式试验一下这个逻辑是否能行得通。

## 2.6 为SKNode类添加SKAction

当然呢我们希望在点击飞机的时候飞机有一些动作出现，那么接下来我们就给点击事件添加一个动作，添加动作需要的类是SKAction，下边我们创建一个动作，这个动作可以让一个精灵节点的实例向上移动。

然后呢，我们用runAction方法将这个动作添加进精灵节点。编译运行，我们就可以看到我们的动作已经实现了。以上就是如何添加SKAction的讲解。

到此为止这一个模块就讲完了，本模块很浅显的创建了一个自己的Sprite Kit工程，主要的目的是让大家熟悉编程环境，同时这个项目本身可以作为一个实验室，比如以后准备写大型游戏的时候有些小的地方不明白，比如Sprite Kit框架更新之后又出了新的方法和功能，都可以在这个项目里单独把功能点拿出来做实验来了解它的细节。

# 三、框架中Node类的各种应用

## 3.1 SKNode类的介绍

大家好，在我们详细了解每个节点的详细用途之前，我先讲解一下SKNode这个基础节点类的知识：

首先是这个类的特性，这个基础类本身虽然是所有具体节点类的父类，但是他本身自己并不承载什么内容渲染，它本身可以是一个容器，我们可以依靠addChild方法给他添加内容。

其次呢，这个不承载内容的类本身也是后很多用途的，这里介绍三个主要的场景

1. 独立类，比如游戏中经常使用的消息提示框我们可以为他们开辟一个最高级的区域
2. 场景分层，假如内容分层特别明显的话，我们可以考虑借助SkNode来实现图层的功能
3. 隐性元素，SKNode本身是有位置坐标的，比如说摄像机这样的隐形参照系，我们也可以使用SKNode

最后讲一下这样做的优点：

1. 可以让我们更加有效地管理场景里的元素，比如批量的隐藏或者删除，又或是统一调整位置的时候比如说屏幕整体的震动效果也会非常方便，
2. 可以实现更加复杂的动作以及物理模拟的效果，这个会在后边的具体实例中讲到。

好，SKNode的基础类特性就讲解到这里，谢谢大家。

# 四、如何向Node类的元素添加Actions

## SKAction类的介绍

大家好，深入学习每个SKAction之前，我们来先了解一下SKAction的基本特征，

首先呢，SKAction本质上定义的是整个游戏画面的一个改动，没有它呢，我们的游戏就是一堆静止的画面，跟普通的应用是没有区别的，有了它呢我们就可以创建丰富的动画来表现游戏的内容。

其次，SKAction总是需要依附于节点实现的，哪怕是播放音乐这种独立的行为，也需要依附于节点来使SKAction本身发挥作用。

最后，SKAction本身是一个符合设计模式中工厂模式的类，他有很多很多方法可以帮我们生成各种SKAction对象的实例，而无需我们自己来手动alloc。

好，SKAction类的介绍就讲到这里谢谢大家。

# 五、搭建游戏场景与场景之间的动态切换

## SKScene类的介绍

大家好，接下来我们来了解一下SKScene类的基础知识，

对于大的框架来讲，SKScene的主要作用是为SKView提供可以渲染的内容，而SKView本身负责管理和调度SKScene。

对于SKScene本身来讲呢，SKScene本身自己的内容组织是一个树状结构，而场景本身是根节点，场景中的内容是按照在在树状结构中的位置按照树算法中的前序遍历的顺序渲染的。

而对于子节点来讲，SKScene类具有完全相同的坐标系统，只不过不同的是中心锚点默认在左下方，

而像xScale，yScale，zRotation这样的位置属性，aplha，hidden，speed这些渲染和动作属性，他跟所有的SKNode一样是可以同时影响到子节点的。

OK，有关SKScene场景类基础知识的介绍就到这里，谢谢大家。