

第四章 游戏引擎概览

本章目录

- 游戏引擎的提出
- 常用游戏引擎
- Cocos2d-x引擎介绍
- Cocos2d-x开发环境配置

上节回顾

- 游戏引擎是一种特殊的软件
- 提供游戏开发时需要的常见功能
 - 通常包含渲染器、2D/3D 图形元素、碰撞检测、物理引擎、声音、控制器支持、动画等
- 提供许多组件
 - 能缩短开发时间
 - 让游戏开发变得更简单



上节回顾

- Cocos2d-x 就是这样一款游戏引擎：
 - 提供了许多易于使用的组件
 - 同时支持移动端和桌面端
 - 通过封装底层图形接口提供了易用的API
 - 降低了游戏开发的门槛
 - 让使用者可以专注于开发游戏
 - 不用关注底层的技术细节
 - 完全开源



本节目标

- 精通Cocos2d-x?

-

很难



保持微笑

- 上手Cocos2d-x?

-

容易



- 先从基本概念，开始 →

开始游戏 >

本节目录

- Cocos2d-x引擎介绍
 - Cocos2d-x 基本概念
 - Cocos2d-x 中的坐标系
 - Cocos2d-x 基础类

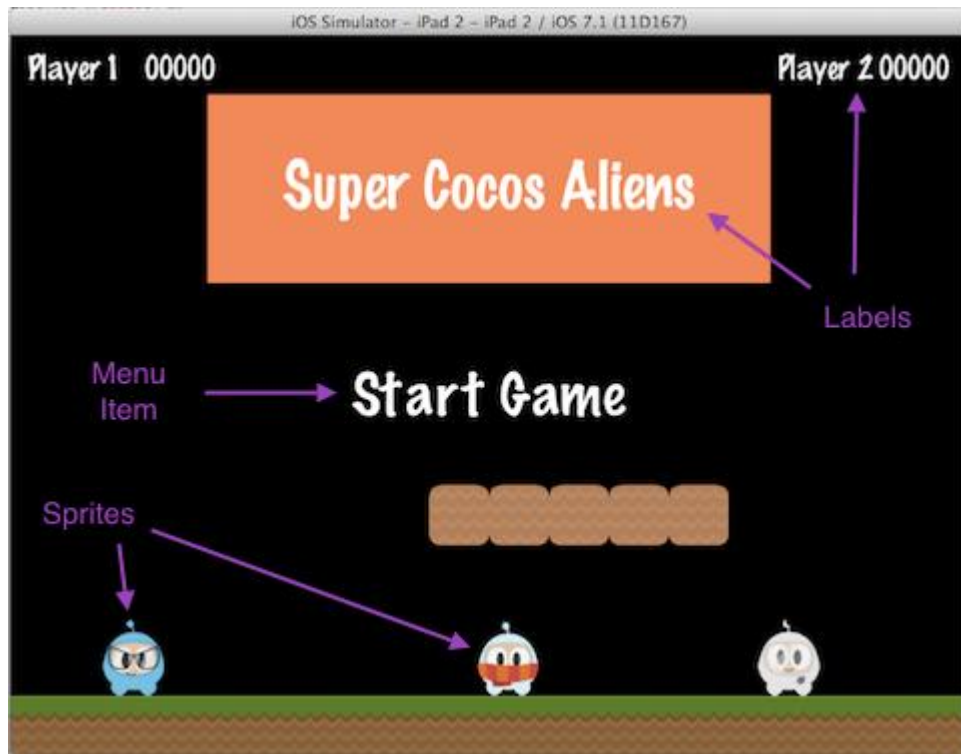
Cocos2d-x 基本概念

- Q: 以下游戏界面包含哪些组件?



Cocos2d-x 基本概念

- 答：菜单(Menu)、菜单项 (Menu Item)、精灵(Sprite)、标签(Label) ...



Cocos2d-x 基本概念

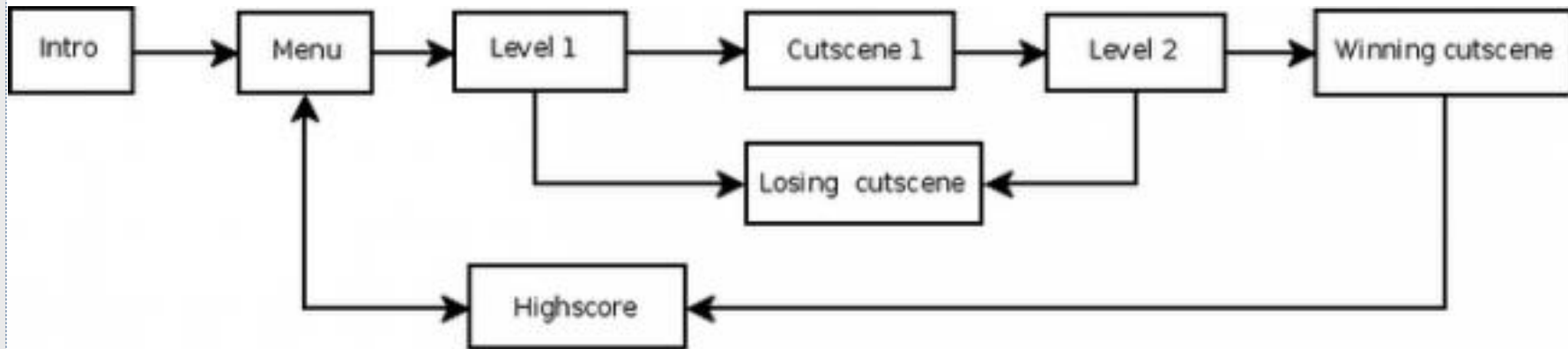
- 导演 (Director)

- 控制场景替换和转换



- 共享的**单例**对象

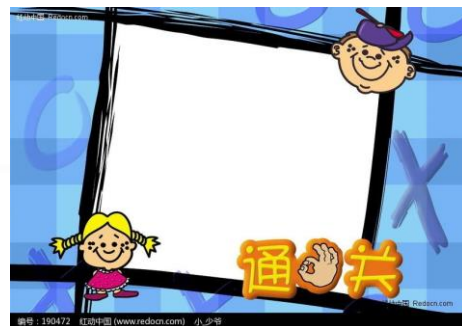
- 可在代码中任何地方**调用**







Cocos2d-x 基本概念

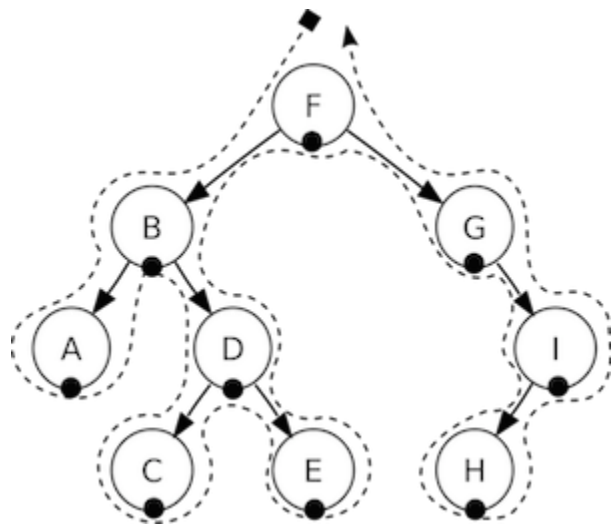
- 场景 (Scene)

请排序



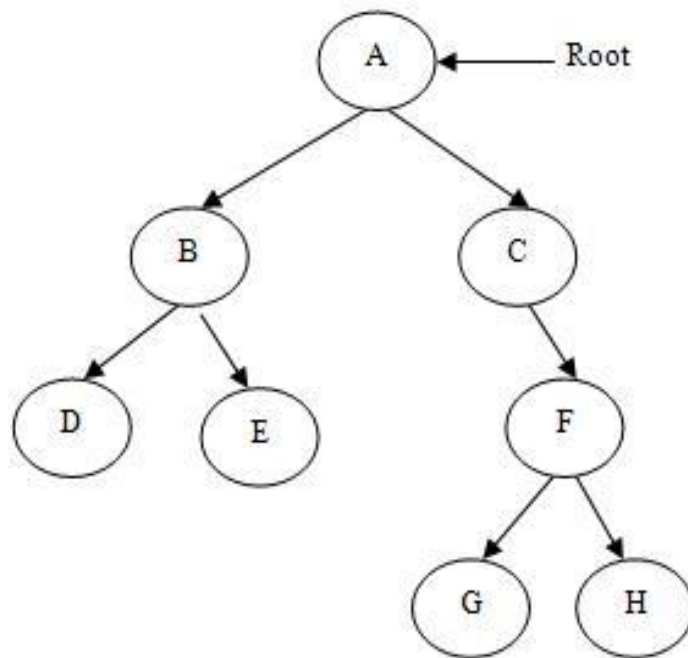
Cocos2d-x 基本概念

- 场景 (Scene)  
- 渲染器 (Renderer)
 - 负责渲染精灵和其它的对象进入屏幕  
- 场景图 (Scene Graph)
 - 安排场景内对象的数据结构
 - 中序遍历 (树)
 - 先遍历左子树
 - 然后是根节点
 - 最后是右子树



Cocos2d-x 基本概念

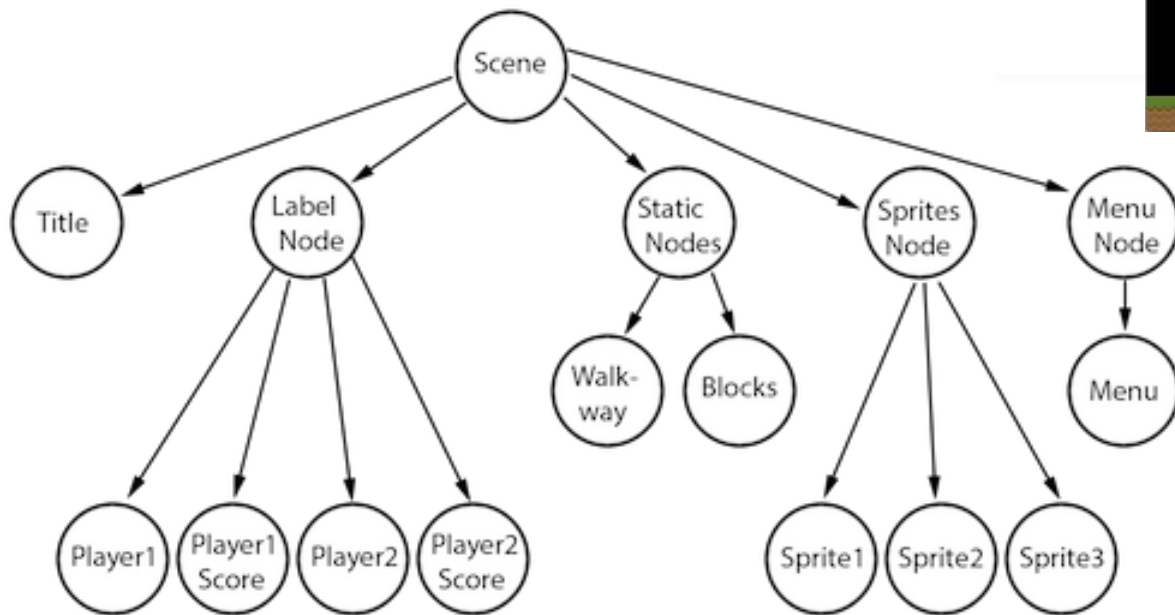
- 场景图 (Scene Graph)



– D,B,E,A,C,G,F,H

Cocos2d-x 基本概念

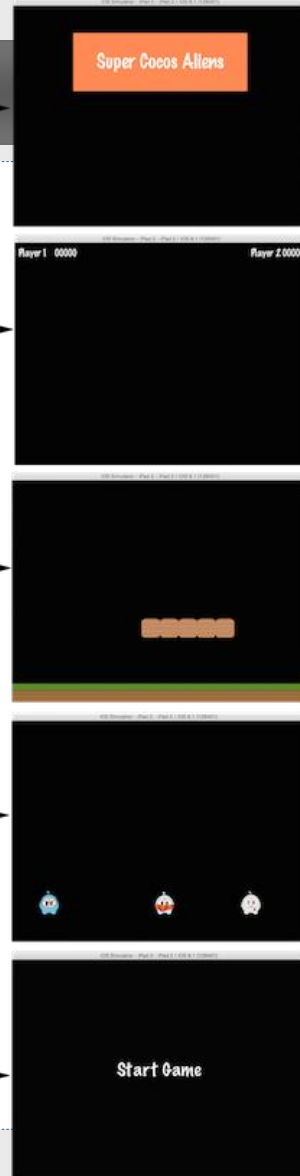
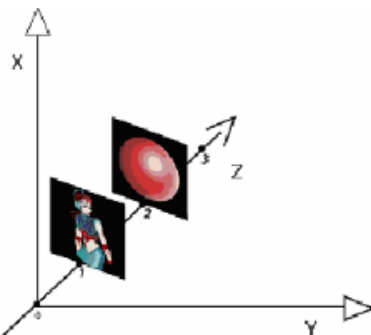
- 场景图 (Scene Graph)



Cocos2d-x 基本概念

- 渲染顺序

- z-order 为负的节点会被放置在左子树
- z-order 非负的节点会被放置在右子树
- 实际开发过程中，可以按照任意顺序添加对象，他们会按照指定的 z-order 自动排序



Cocos2d-x 基本概念



• 构建场景图

1

```
// Adds a child with the z-order of -2, that means  
// it goes to the "left" side of the tree (because it is negative)  
scene->addChild(title_node, -2);
```

z-order

2

```
// When you don't specify the z-order, it will use 0  
scene->addChild(label_node);
```

3

```
// Adds a child with the z-order of 1, that means  
// it goes to the "right" side of the tree (because it is positive)  
scene->addChild(sprite_node, 1);
```

Cocos2d-x 基本概念

- 精灵 (Sprite)

- 是玩家在屏幕上移动的对象，**能被控制**。
- 能通过改变自身的**属性**：角度、位置、缩放、颜色等，变成**可控制**动画的 2D 图像。

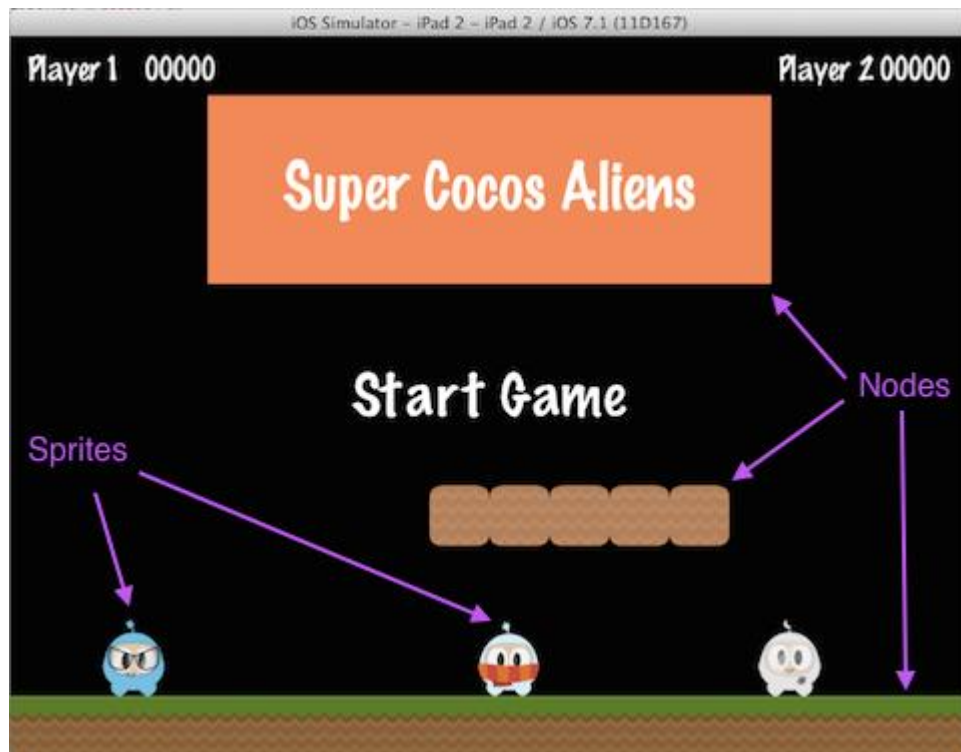
- 每个游戏中都有精灵吗？

- 每个图形对象都是精灵吗？



Cocos2d-x 基本概念

- 精灵 (Sprite)

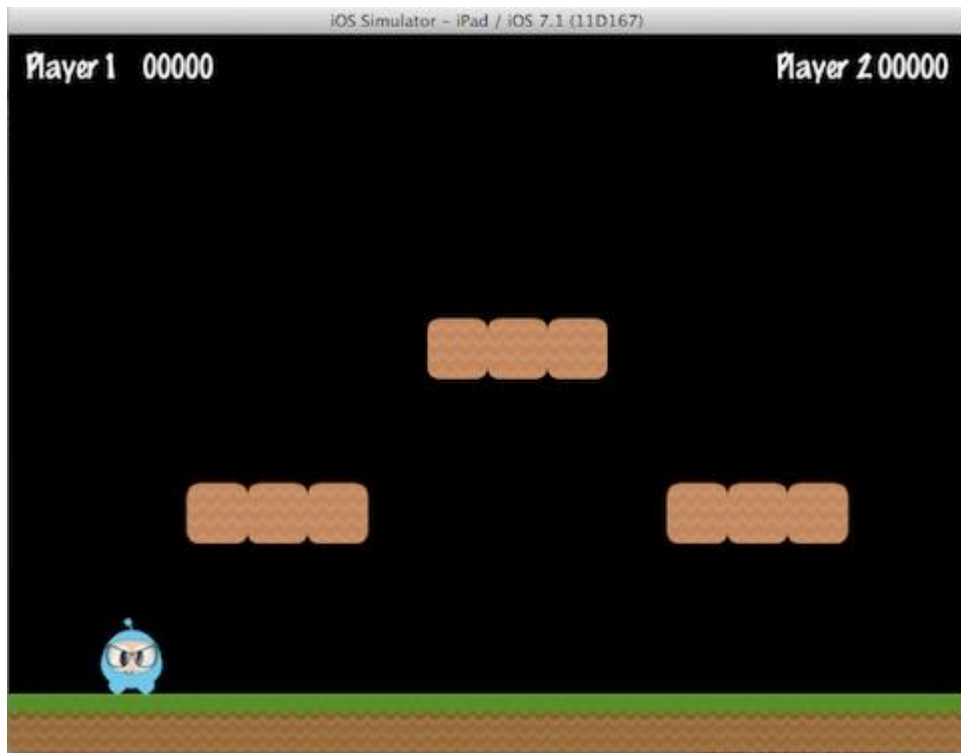


Cocos2d-x 基本概念

- 精灵的创建

```
// This is how to create a sprite
```

```
auto mySprite = Sprite::create("mysprite.png");
```



Cocos2d-x 基本概念

- 精灵的属性设置 设置位置 `mySprite->setPosition(Vec2(500, 0));`



Cocos2d-x 基本概念

- 精灵的属性设置

设置旋转角度

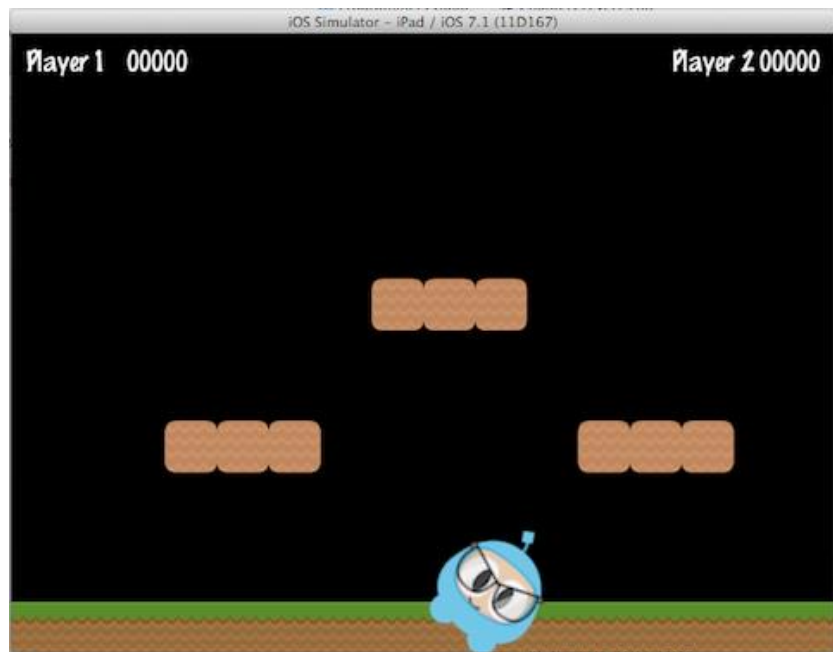
```
mySprite->setRotation(40);
```



Cocos2d-x 基本概念

- 精灵的属性设置

设置缩放比例 `mySprite->setScale(2.0);`



本节目录

- Cocos2d-x引擎介绍
 - Cocos2d-x 基本概念
 - Cocos2d-x 中的坐标系
 - Cocos2d-x 基础类

Cocos2d-x中的坐标系

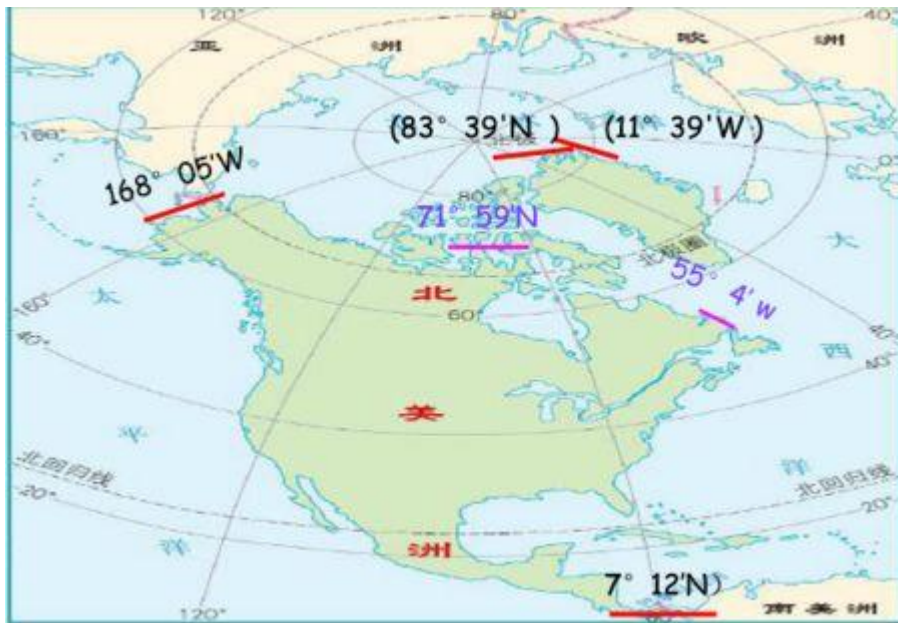
- 标准屏幕坐标系
 - 原点：屏幕左上角
 - X轴：正方向向右
 - Y轴：正方向向下
- Cocos2d-x坐标系
 - 原点：屏幕左下角
 - X轴：正方向向右
 - Y轴：正方向向上



OpenGL / OpenGL ES

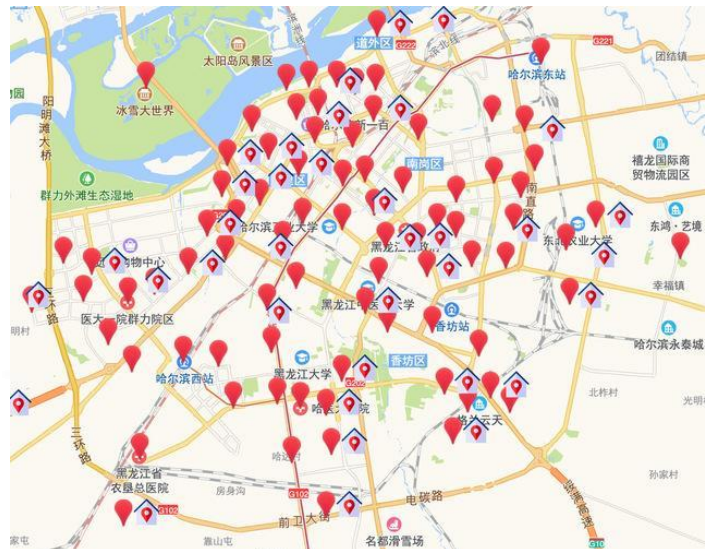
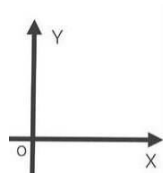
Cocos2d-x中的坐标系

- 世界坐标系
 - 也叫绝对坐标系
 - 提供其它坐标系所需的参考标准



Cocos2d-x中的坐标系

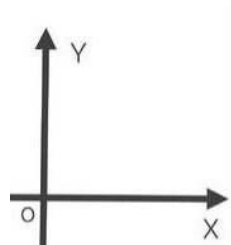
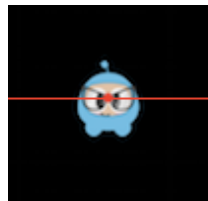
- 本地坐标系
 - 也叫局部坐标系
 - 每个节点的独立坐标系
 - 相对于父对象的坐标系
 - 原点：父对象左下角
 - X轴：正方向向右
 - Y轴：正方向向上



```
auto layer = new Layer();  
auto player = Sprite::create("player.png");  
layer->addChild(player, 1);
```

Cocos2d-x中的坐标系

- 锚点 (Anchor Point)
 - 节点(Node)对象计算坐标位置时的**基准点**
 - 所有的节点对象都有锚点
 - 只有节点对象使用贴图的情况下，锚点才有意义
 - 节点对象锚点的默认值：(0.5, 0.5)
 - 范围[0, 1]，表示的是一个**乘数因子**
 - 使用的坐标系以**左下角**为原点 (0, 0)
 - 表示的是**相对坐标**



Cocos2d-x中的坐标系

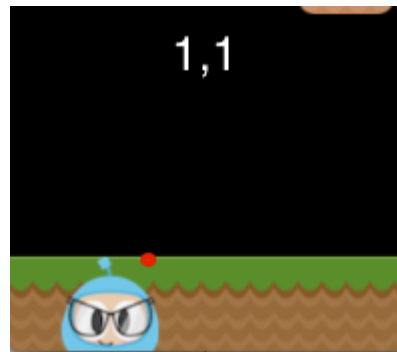
- 锚点 (Anchor Point)

(1,1)

- 以刚才展示的精灵为例:

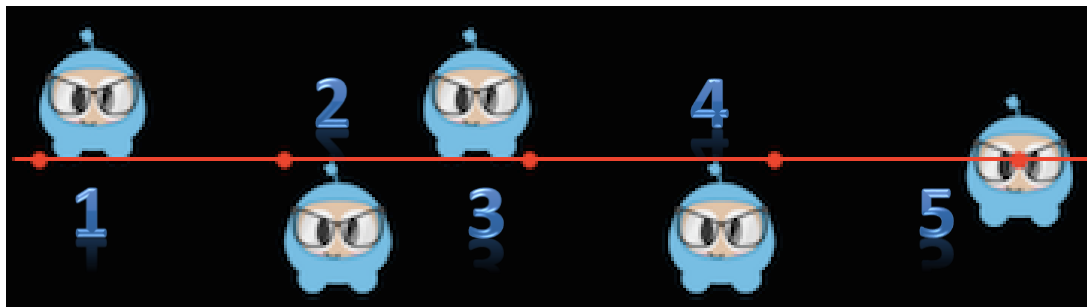
- `mySprite->setPosition(Vec2(200, 100));`

- `mySprite->setAnchorPoint(Vec2(0, 0));`



Cocos2d-x中的坐标系

- 锚点 (Anchor Point)



```
mySprite->setAnchorPoint(1, 1);
```

```
mySprite->setAnchorPoint(0, 1);
```

```
mySprite->setAnchorPoint(0, 0);
```

```
mySprite->setAnchorPoint(0.5, 0.5);
```

```
mySprite->setAnchorPoint(1, 0);
```



Cocos2d-x中的坐标系

- 锚点 (Anchor Point)

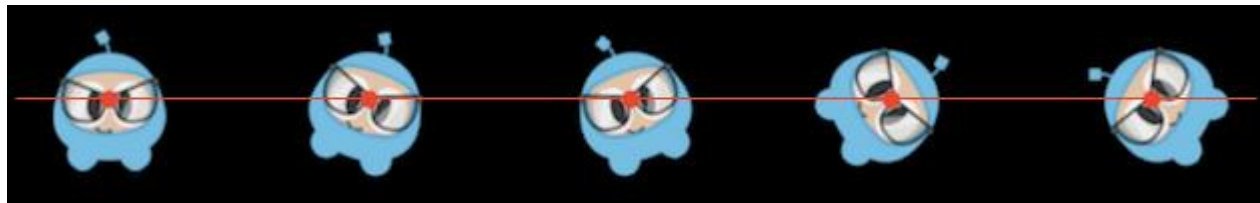
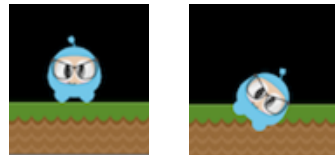


- 当设置精灵位置时，主要使用 setPosition () 方法
- 只有 想改变精灵与基准坐标点的相对位置 时，才考虑使用 setAnchorPoint () 方法

— 锚点设置不同，是否会影响精灵的缩放、旋转、倾斜、颜色、透明度属性？

Cocos2d-x中的坐标系

- 锚点 & 精灵的旋转



1

2

3

4

5

```
mySprite->setRotation(-20.0f);
```

```
mySprite->setRotation(60.0f);
```

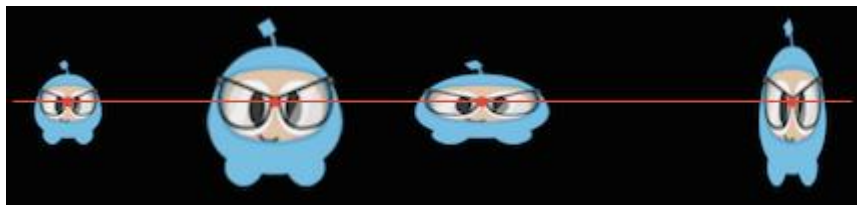
```
mySprite->setRotation(20.0f);
```

```
mySprite->setRotation(-60.0f);
```



Cocos2d-x中的坐标系

- 锚点 & 精灵的缩放



1

2

3

4

```
mySprite->setScale(2.0);
```

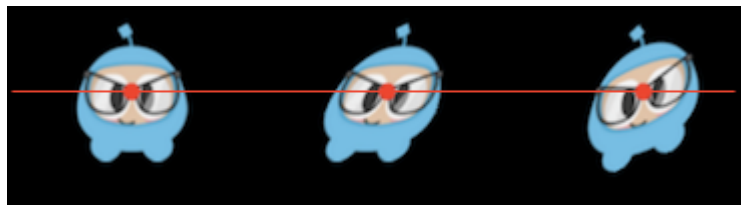
```
mySprite->setScaleY(2.0);
```

```
mySprite->setScaleX(2.0);
```



Cocos2d-x中的坐标系

- 锚点 & 精灵的倾斜



1

2

3

```
mySprite->setSkewX(20.0f);
```

```
mySprite->setSkewY(20.0f);
```

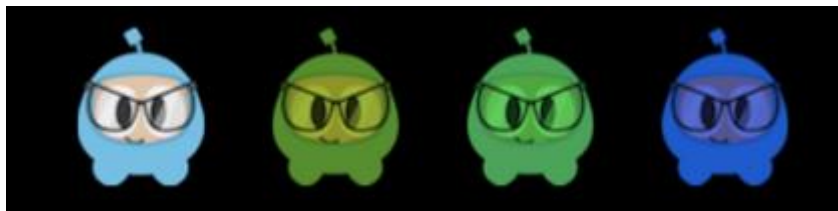


YES



Cocos2d-x中的坐标系

- 锚点 & 精灵的颜色



```
// set the color by passing in a pre-defined Color3B object.
```

```
mySprite->setColor(Color3B::WHITE);
```

```
// Set the color by passing in a Color3B object.
```

```
mySprite->setColor(Color3B(255, 255, 255)); // Same as Color3B::WHITE
```

Cocos2d-x中的坐标系

- 锚点 & 精灵的透明度



Opaque

1

2

3

4

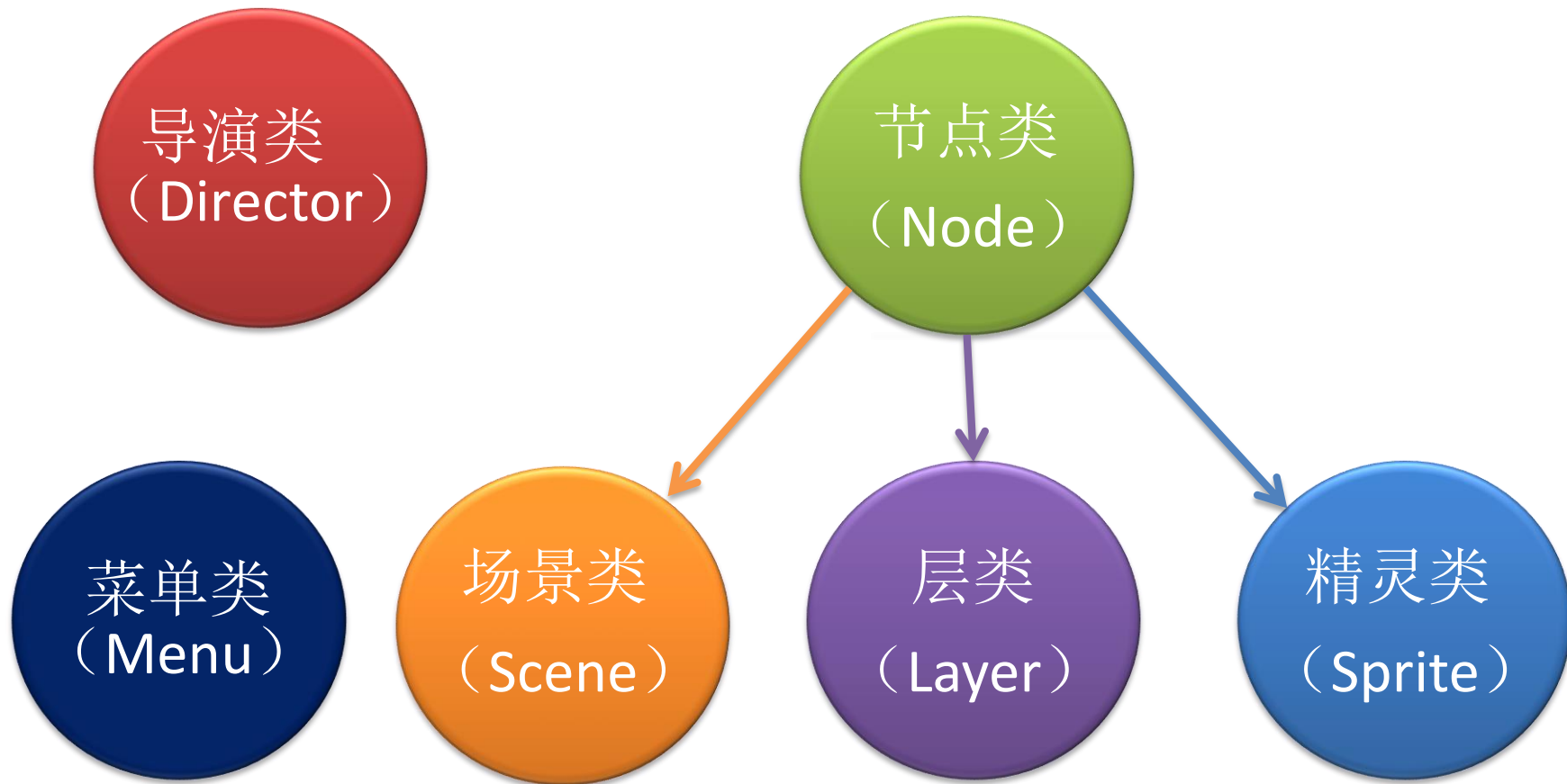


```
// Set the opacity to 30, which makes this sprite 11.7% opaque.  
// (30 divided by 256 equals 0.1171875...)  
mySprite->setOpacity(30);
```

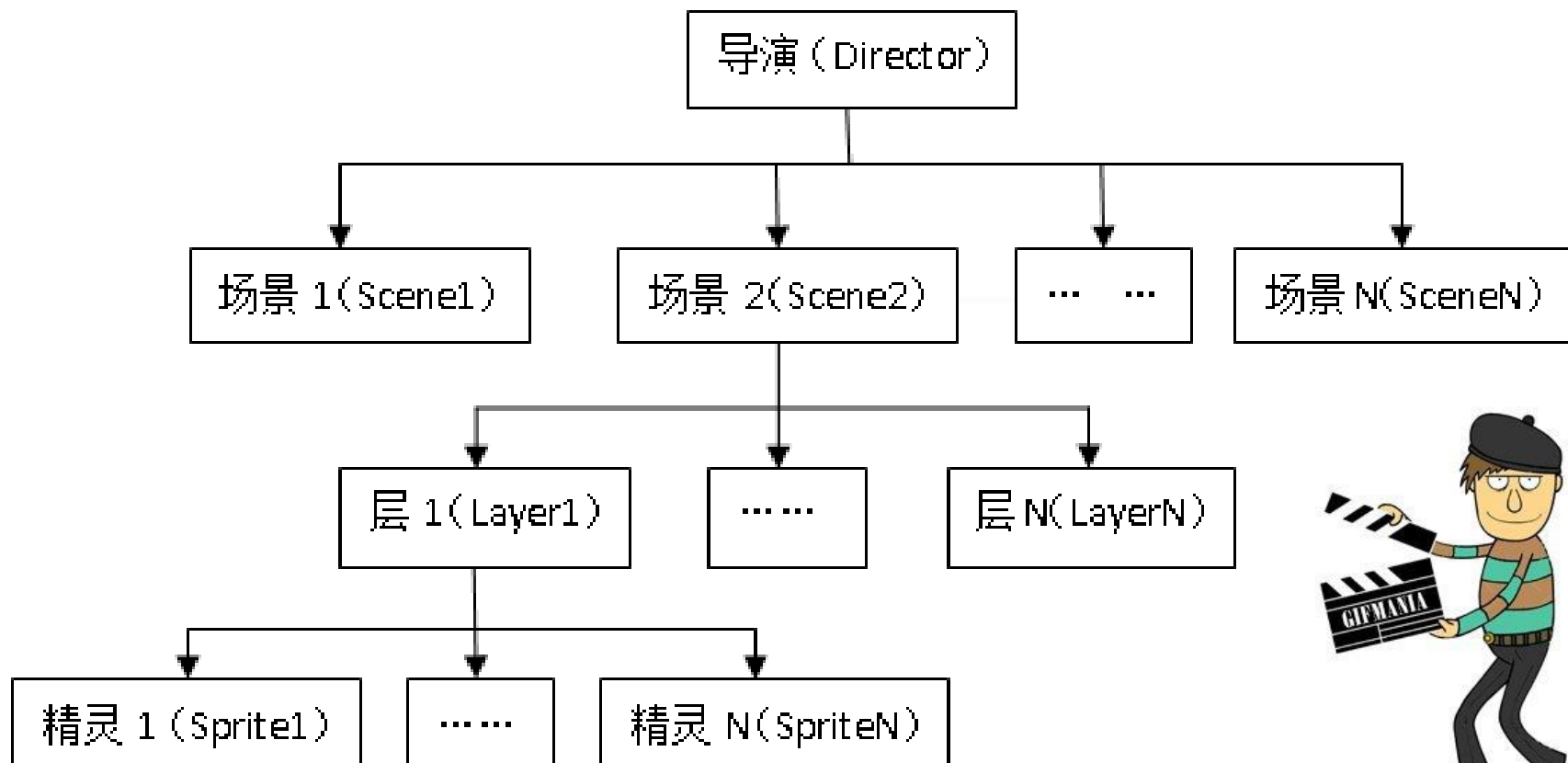
本节目录

- Cocos2d-x引擎介绍
 - Cocos2d-x 基本概念
 - Cocos2d-x 中的坐标系
 - Cocos2d-x 基础类

Cocos2d-x基础类



Cocos2d-x基础类



Cocos2d-x基础类

- 导演类 (Director)
 - 游戏的“**总指挥**”、引擎的**控制核心**
 - OpenGL ES初始化
 - 场景的转换
 - 游戏的暂停、继续
 - 世界坐标和GL坐标之间的切换
 - 对节点（游戏元素）的控制
 - 游戏数据的保存、调用
 - 获取屏幕尺寸



Cocos2d-x基础类

- 导演类 (Director)

- 采取单例模式

- 通过 `Director::getInstance()` 获取实例

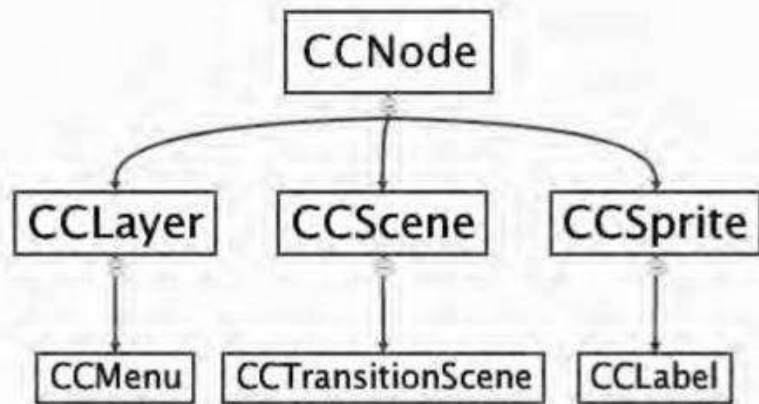
解决方案'MyGame' (6 个项目)

- ▷ libbox2d
- ▷ libbullet
- ▷ libcocos2d
- ▷ librecast
- ▷ libSpine
- ▲ MyGame
 - ▷ 引用
 - ▷ 外部依赖项
 - ▷ resource
 - ▲ src
 - ▷ AppDelegate.cpp
 - ▷ AppDelegate.h
 - ▷ HelloWorldScene.cpp
 - ▷ HelloWorldScene.h
 - ▲ win32
 - ▷ main.cpp
 - ▷ main.h

```
56 bool AppDelegate::applicationDidFinishLaunching() {
57     // initialize director
58     auto director = Director::getInstance();
59     auto glview = director->getOpenGLView();
60     if(!glview) {
61         #if (CC_TARGET_PLATFORM == CC_PLATFORM_WIN32) || (CC_T
62             glview = GLViewImpl::createWithRect("MyGame",
63         #else
64             glview = GLViewImpl::create("MyGame");
65         #endif
66         director->setOpenGLView(glview);
67     }
```

Cocos2d-x基础类

- 节点类 (Node)
 - 游戏中大部分类继承自Node类
 - Node类继承自Ref类 (所有类的基类)
 - 与渲染相关的类几乎都继承自Node类
 - 场景类 (Scene)
 - 层类 (Layer)
 - 精灵类 (Sprite)
 - 菜单类 (Menu)



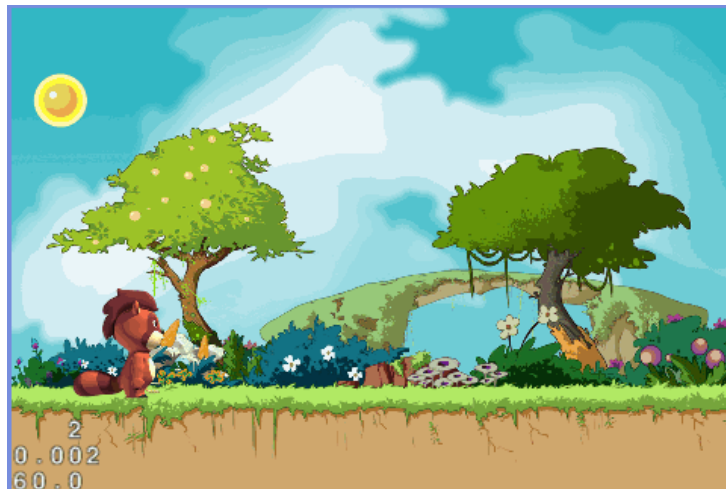
Cocos2d-x基础类

- 场景类 (Scene)
 - 构成游戏的一个个界面、关卡与板块
 - 提供场景切换效果
 - 旋转、翻页、淡入淡出



Cocos2d-x基础类

- 层类 (Layer)
 - 游戏元素的容器
 - 处理玩家事件响应
 - 触摸事件
 - 鼠标事件
 - 加速度计事件
 - 键盘输入
- 精灵类 (Sprite)



Cocos2d-x基础类

- 构建一个简单场景

```
auto dirs = Director::getInstance();  
Size visibleSize = dirs->getVisibleSize();
```

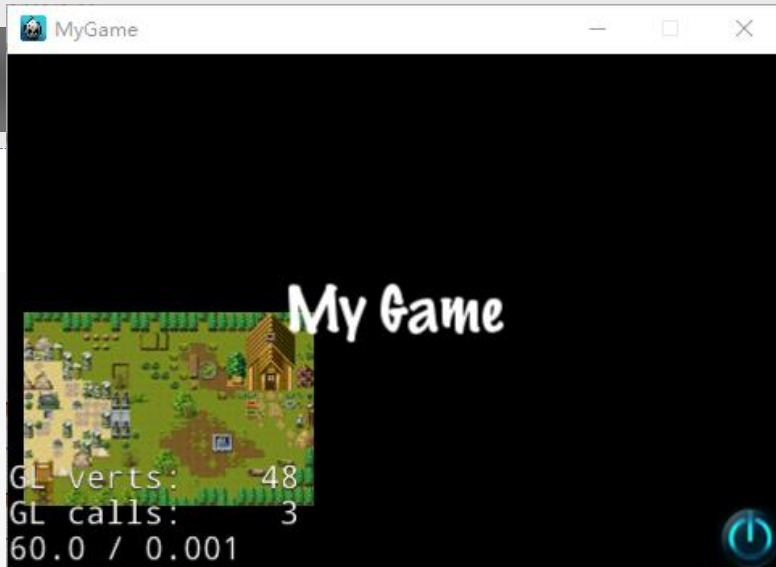
```
auto myScene = Scene::create();
```

```
auto label1 = Label::createWithTTF("My Game", "Marker Felt.ttf", 36);  
label1->setPosition(Vec2(visibleSize.width / 2, visibleSize.height / 2));
```

```
myScene->addChild(label1);
```

```
auto sprite1 = Sprite::create("mysprite.png");  
sprite1->setPosition(Vec2(100, 100));
```

```
myScene->addChild(sprite1);
```



```
designResolutionSize = cocos2d::Size(480, 320);
```

小结

- Cocos2d-x基本概念
 - 组件构成
 - 导演、场景、精灵
- Cocos2d-x坐标系
 - 各种坐标系
 - 锚点
- Cocos2d-x基础类
 - 导演、节点、场景、层、精灵、菜单