

Cocos2dx 游戏开发教程

by 陈祥麟



目录 / contents

01

游戏的运行原理

02

Cocos2d-x安装配置

03

Hello world



游戏运行方式:

- 渲染画面
- 事件触发



渲染画面:



COCOS2D X

渲染：

用软件从模型生成图像的过程。模型是用语言或者数据结构进行严格定义的三维物体或虚拟场景的描述，它包括几何、视点、纹理、照明和阴影等信息。图像是数字图像或者位图图像。

```
while(!exit) {  
    DrawGame();  
}
```



帧率 (FPS) :

- 每秒刷新页面的次数
- 延时 (sleep())
- 30 ~ 60 (根据游戏类型自己设定)



事件：

事件是可以被控件识别的操作，如按下确定按钮，选择某个单选按钮或者复选框。每一种控件有自己可以识别的事件，如窗体的加载、单击、双击等事件，操控人物前进，后退，攻击等等。

```
event() {  
    attack();  
}
```



游戏引擎：

游戏引擎是指一些已编写好的可编辑电脑游戏系统或者一些交互式实时图像应用程序的核心组件。这些系统为游戏设计者提供各种编写游戏所需的各种工具，其目的在于让游戏设计者能容易和快速地做出游戏程式而不用由零开始。大部分都支持多种操作平台，如Linux、Mac OS X、微软Windows。游戏引擎包含以下系统：渲染引擎（即“渲染器”，含二维图像引擎和三维图像引擎）、物理引擎、碰撞检测系统、音效、脚本引擎、电脑动画、人工智能、网络引擎以及场景管理。



Why cocos2d-x?

目前，移动游戏引擎主要有Cocos2d-x、Unity3D、FlashAIR、Unreal、Corona等。但总体来讲，行业目前首选的游戏引擎，主要都集中在Cocos2d-x与Unity3D之上。

从全球市场份额数据来看，主要覆盖中端市场的Unity相对领先，Cocos2d-x则主要占据高端与低端市场，约占**1/4**市场。但值得注意的是另一个数据是，在中国，Cocos2d-x则相对领先。目前，在中国的2D手机游戏开发中，Cocos2d-x引擎的份额超过**70%**。

根据触控科技数据，国内现有**45**款月收入超千万手游中，**30**个基于Cocos2d-x开发，2013年手机游戏产业的22起手游并购案中，收购股权大于51%有**20**起，其中**13**起的代表游戏均基于Cocos2d-x开发。



COCOS2D-X

Cocos2d-x是一个MIT许可证下发布的开源的移动2D游戏框架，它有以下特征:



- **跨平台**: iOS, Android, Win32, WP...
- **多语言**: C++, Javascript (JSB), LUA
- **开源免费**: MIT License
- **简单上手，运行高效、灵活，且功能强大**
- **各类编辑器**: CocoStudio, Particle Designer, Tile Map...
- **第三方插件**: Plugin-X, 社交，广告，支付...
- **详细的文档**: 官方wiki <http://www.cocos2d-x.org/wiki>
或者GitHub Doc <https://github.com/cocos2d/cocos-docs/>
- **使用的人多，遇见了问题便能够更加方便地上网查询解决方案**



安装cocos2d-x:

- 安装python2.7
- 安装cocos2d-x 3.x
- 新建Hello World

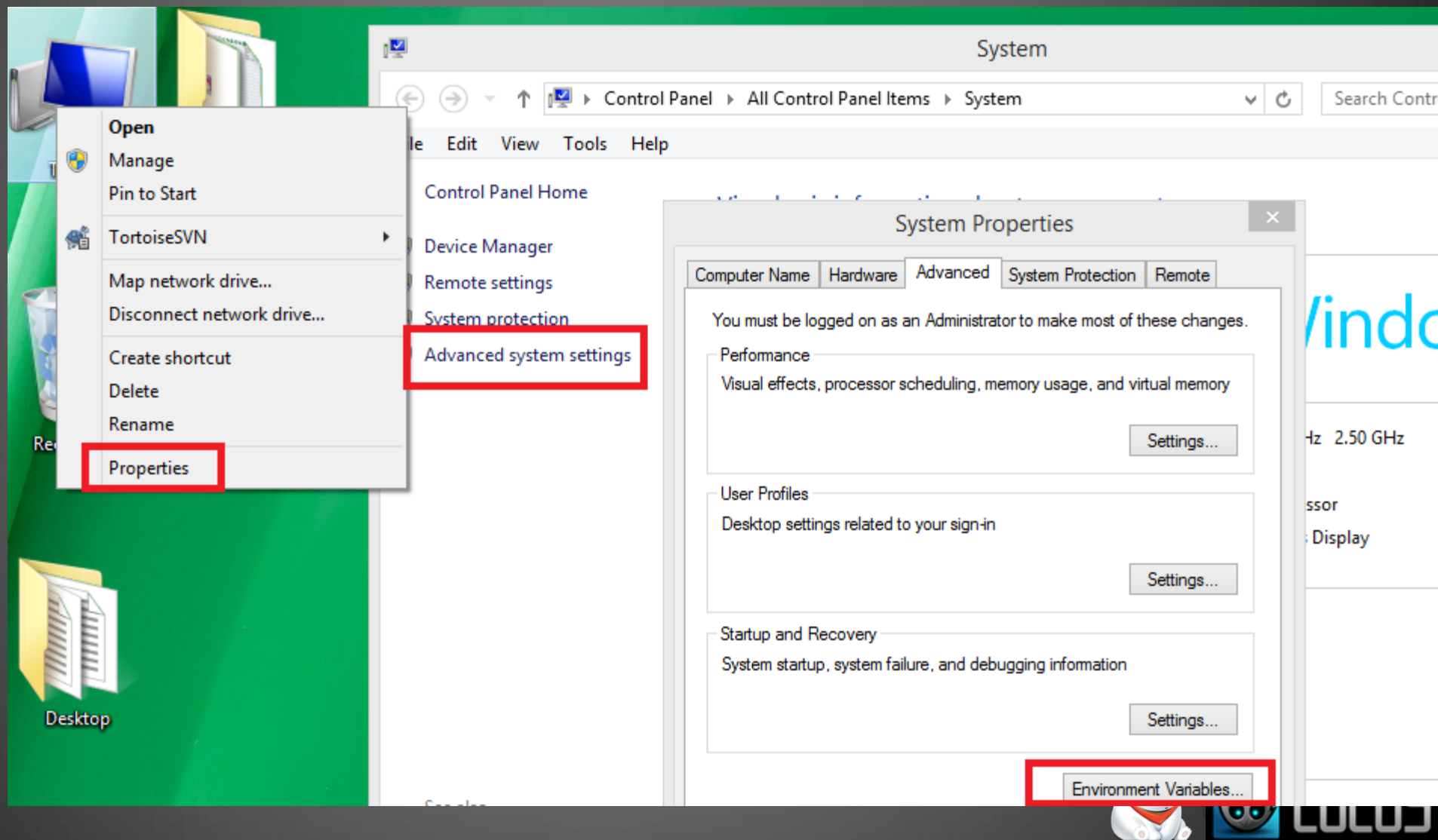


Python2.7 :

- <https://www.python.org/downloads/>
- 安装
- 设置环境变量

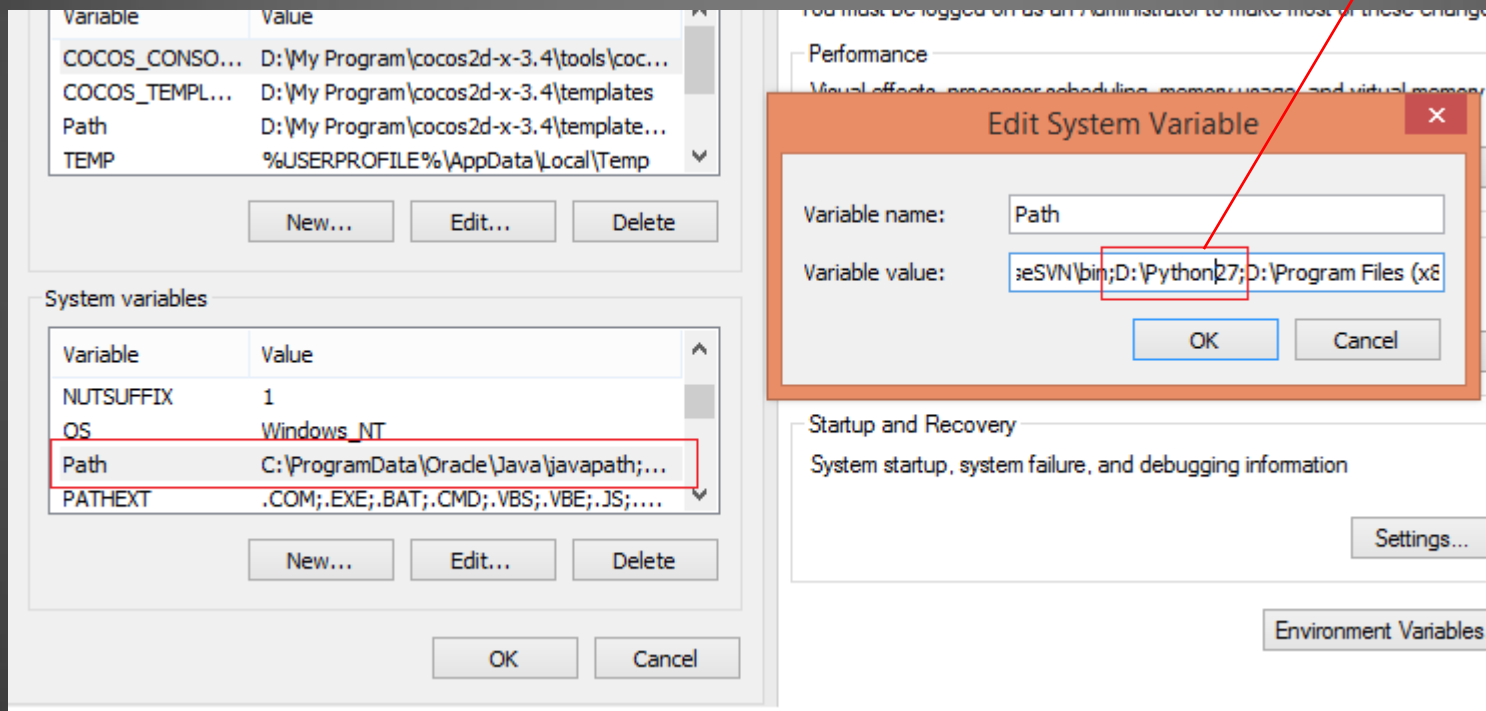


环境变量：



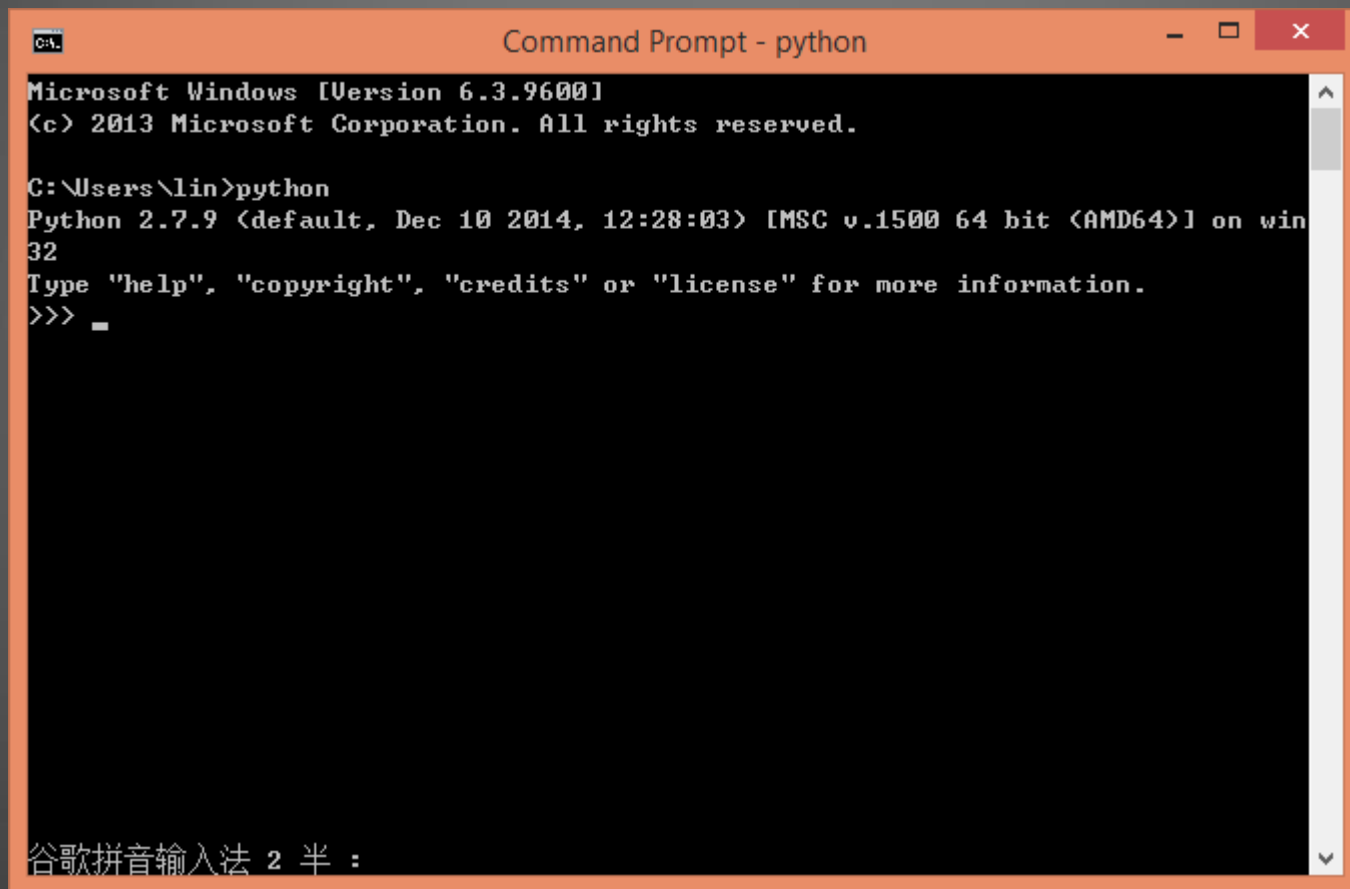
环境变量：

你的安装目录，注意分号



测试：

控制台输入命令：python



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\lin>python
Python 2.7.9 (default, Dec 10 2014, 12:28:03) [MSC v.1500 64 bit (AMD64)] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```



COCOS2D X

Cocos2d-x :

- <http://cn.cocos2d-x.org/download/>
- 解压缩
- 在当前目录打开命令行，输入python setup.py



Cocos2d-x :

```
C:\Windows\system32\cmd.exe

D:\My Program\cocos2d-x-3.4>python setup.py

Setting up cocos2d-x...
->Check environment variable COCOS_CONSOLE_ROOT
  ->Search for environment variable COCOS_CONSOLE_ROOT...
    ->COCOS_CONSOLE_ROOT is found : D:\My Program\cocos2d-x-3.4\tools\cocos2d-console\bin

->Check environment variable COCOS_TEMPLATES_ROOT
  ->Search for environment variable COCOS_TEMPLATES_ROOT...
    ->COCOS_TEMPLATES_ROOT is found : D:\My Program\cocos2d-x-3.4\templates

->Configuration for Android platform only, you can also skip and manually edit your environment variables

->Check environment variable NDK_ROOT
  ->Search for environment variable NDK_ROOT...
    ->NDK_ROOT not found

  ->Search for command ndk-build in system...
    ->Command ndk-build not found

  ->Please enter the path of NDK_ROOT (or press Enter to skip):
->Check environment variable ANDROID_SDK_ROOT
  ->Search for environment variable ANDROID_SDK_ROOT...
谷歌拼音输入法 2 半 :
```

直接不断地enter跳过,但是如果要编译到安卓,需要NDK, SDK等,便需要在这里进行设置



Cocos2d-x :

- 进入build目录，打开cocos2d-win8.1-universal.sln
- 设置cpptest.windows为启动项（ test中有几乎所有的类的使用 ）
- 生成解决方案



新建项目：

- 控制

p cc

d D:

```
Command Prompt
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\lin>cocos new demopackage -p com.cc.c -l cpp -d D:\cocosexample
Running command: new
> Copy template into D:\cocosexample\demopackage
> Copying cocos2d-x files...
> Rename project name from 'HelloCpp' to 'demopackage'
> Replace the project name from 'HelloCpp' to 'demopackage'
> Replace the project package name from 'org.cocos2dx.hellocpp' to 'com.cc.c'
> Replace the mac bundle id from 'org.cocos2dx.hellocpp' to 'com.cc.c'
> Replace the ios bundle id from 'org.cocos2dx.hellocpp' to 'com.cc.c'
C:\Users\lin>
```

名称) -

类型) -

谷歌拼音输入法 2 半 :



文件目录：

| 文件名 | 说明 |
|------------------|--|
| build | 官方编译的项目解决方案。 集中放在这个文件中，如cocos2d-win32.vc2012.sln、Debug.win32、win32-msvc-2012-x86.cmd等。 这些在2.x版本是直接放在根目录的。 |
| cocos | 集中了Cocos2d-x开发中使用的 所有源文件 。 除了Cocos2d-x引擎核心部分，还包含声音引擎、物理引擎、网络、GUI等。 |
| extensions | 扩展目录。 包括一些2.5D特效，网络控制，以及一些GUI等，当要用到这些扩展时： <pre>#include "cocos-ext.h" using namespace cocos2d::extension</pre> |
| external | 第三方目录库。 主要包含了Box2D、Chipmunk物理引擎，以及Sqlite3轻型数据库等。 |
| tests | 官方样例。 包含了官方的C++、lua例子。其中cpp-empty-test就是最简单的例子。 另外也包含了一个 <code>cpp-tests</code> 项目，里面涵盖了cocos2dx引擎所有类的使用，所以很重要！ |
| templates | 模板目录。 提供了各种平台及IDE下创建的cocos2dx新项目的项目模板。 |
| tools | 工具目录。 提供了各个平台下，创建cocos2dx新项目的脚本工具。 其中： <code>\tools\cocos2d-console\bin\cocos.py</code> 就是用来创建cocos2dx新项目的。 |
| download-deps.py | 自动下载Cocos2d-x所需的第三方插件（如果有的话） |
| setup.py | 配置Cocos2d-x的环境变量的脚本。 |



COCOS2D-X

COCOS :

| 文件名 | 说明 |
|-------------------|---|
| base | Cocos2d-x引擎的核心部分之一。 存放一些基础类，如Ref、Director、Vector、Map、Value等。 |
| 2d | Cocos2d-x引擎的核心部分之一。 存放一些核心部分的类，如Sprite、Layer、Label、Menu等。 |
| 3d | 新增了对部分3D的支持。 有Sprite3D、Animate3D等，呵呵，可能将来要出cocos3dx了吧。 |
| math | 存放引擎中与数学相关的类文件。 如坐标类Vec2.h、几何类CCGeometry.h（定义了Rect、Size）等。 |
| renderer | 引擎的渲染文件。 Texture2D、Renderer、Shader等。 |
| deprecated | 存放2.x中即将被废弃的规则，目的是为了兼容2.x。 例如，使用typedef来对CC的兼容（如：typedef Label CCLabel）。 |
| platform | 不同平台下的入口类。 CCApplication.h、CCGLView.h、CCGL.h、CCStdC.h |
| audio | 声音引擎相关源文件。 在2.x里是放在根目录的CocosDenshion文件夹。 |
| physics | 物理引擎相关源文件。 chipmunk。 |
| network | 网络相关相关源文件。 HttpClient、WebSocket。 |
| ui | GUI相关源文件。 |
| editor-support | 对编辑器的支持。 cocosbuilder、Cocos Studio、Spine。 |
| scripting | lua脚本相关源文件。 |
| cocos2d.h | Cocos2d-x引擎核心头文件，包含了引擎中的所有类。 可以打开它，查看引擎的所有源文件所在位置。 |



Hello World:

- 游戏的生命周期
- Hello world
- 作业



打开新建的项目：

需要完成的程序代码

项目工程文件

引用的资源

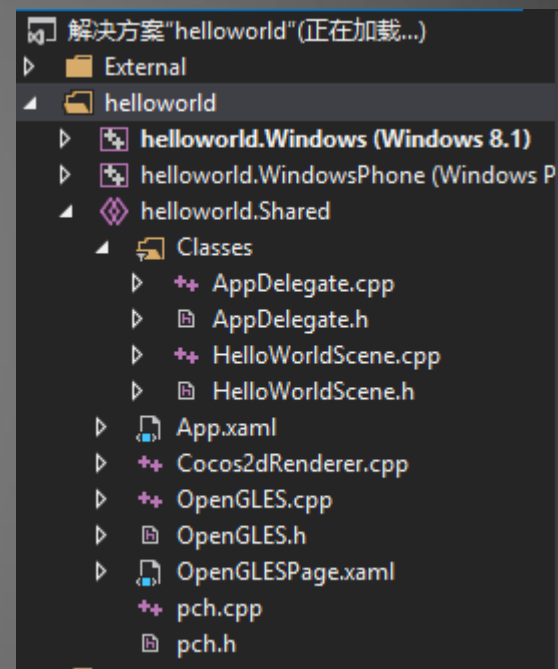
| Name | Date modified | Type | Size |
|-----------------------|-----------------|---------------|------|
| Classes | 2015/4/23 16:25 | File folder | |
| cocos2d | 2015/4/23 16:26 | File folder | |
| proj.android | 2015/4/23 16:25 | File folder | |
| proj.ios_mac | 2015/4/23 16:26 | File folder | |
| proj.linux | 2015/4/23 16:25 | File folder | |
| proj.win8.1-universal | 2015/4/23 18:33 | File folder | |
| proj.win32 | 2015/4/23 16:26 | File folder | |
| proj.wp8-xaml | 2015/4/23 16:26 | File folder | |
| Resources | 2015/4/23 16:25 | File folder | |
| .cocos-project.json | 2015/4/23 16:26 | JSON File | 1 KB |
| CMakeLists.txt | 2015/4/23 16:26 | Text Document | 5 KB |



COCOS2D X

项目结构:

| Name | Date modified | Type | Size |
|--------------------|-----------------|-----------------------|------------|
| App.Shared | 2015/4/23 16:26 | File folder | |
| App.Windows | 2015/4/23 16:26 | File folder | |
| App.WindowsPhone | 2015/4/23 16:26 | File folder | |
| Debug | 2015/4/23 16:26 | File folder | |
| ipch | 2015/4/23 16:27 | File folder | |
| helloworld.sdf | 2015/4/23 18:33 | SQL Server Comp... | 187,264 KB |
| helloworld.sln | 2015/4/23 16:26 | Microsoft Visual S... | 14 KB |
| helloworld.v12.suo | 2015/4/23 18:33 | Visual Studio Solu... | 31 KB |



了解AppDelegate, HelloWorldScene



AppDelegate :

游戏启动完成

应用程序进入后台

游戏从后台恢复

```
/**
 *brief   Implement Director and Scene init code here.
 *return true   Initialize success, app continue.
 *return false  Initialize failed, app terminate.
 */
virtual bool applicationDidFinishLaunching();

/**
 *brief   The function be called when the application enter background
 *param   the pointer of the application
 */
virtual void applicationDidEnterBackground();

/**
 *brief   The function be called when the application enter foreground
 *param   the pointer of the application
 */
virtual void applicationWillEnterForeground();
```



AppDelegate :

创建导演

展示当前帧率

设置帧率

创建并设置初始场景

```
bool AppDelegate::applicationDidFinishLaunching() {  
    // initialize director  
    auto director = Director::getInstance();  
    auto glview = director->getOpenGLView();  
    if(!glview) {  
        glview = GLViewImpl::create("My Game");  
        director->setOpenGLView(glview);  
    }  
  
    // turn on display FPS  
    director->setDisplayStats(true);  
  
    // set FPS. the default value is 1.0/60 if you don't call this  
    director->setAnimationInterval(1.0 / 60);  
  
    // create a scene. it's an autorelease object  
    auto scene = HelloWorld::createScene();  
  
    // run  
    director->runWithScene(scene);  
  
    return true;  
}
```



COCOS2D X

helloworld :

场景

初始化helloworld

退出按钮的函数

宏，创建helloworld对象

```
#include "cocos2d.h"

class HelloWorld : public cocos2d::Layer
{
public:
    // there's no 'id' in cpp, so we recommend returning the class instance pointer
    static cocos2d::Scene* createScene();

    // Here's a difference. Method 'init' in cocos2d-x returns bool, instead of returning
    virtual bool init();

    // a selector callback
    void menuCloseCallback(cocos2d::Ref* pSender);

    // implement the "static create()" method manually
    CREATE_FUNC(HelloWorld);
};
```



helloworld :

在这段代码中，首先利用 `Scene::create` 方法创建了一个空场景，然后利用 `Hello world::create` 方法创建一个 `HelloWorld` 层的实例，最后调用 `scene` 对象的 `addChild` 方法来把创建的层添加到场景之中。

```
Scene* HelloWorld::createScene()  
{  
    // 'scene' is an autorelease object  
    auto scene = Scene::create();  
    // 'layer' is an autorelease object  
    auto layer = HelloWorld::create();  
    // add layer as a child to scene  
    scene->addChild(layer);  
    // return the scene  
    return scene;  
}
```



COCOS2D X

helloworld :

```
// add a "close" icon to exit the progress. it's an autorelease object
auto closeItem = MenuItemImage::create(
    "CloseNormal.png",
    "CloseSelected.png",
    CC_CALLBACK_1 (HelloWorld::menuCloseCallback, this));

closeItem->setPosition(Vec2(origin.x + visibleSize.width - closeItem->getContentSize().width/2 ,
    origin.y + closeItem->getContentSize().height/2));

// create menu, it's an autorelease object
auto menu = Menu::create(closeItem, NULL);
menu->setPosition(Vec2::ZERO);
this->addChild(menu, 1);
```

z坐标，其值越大，越靠前

来自resource

在这段代码中，创建了一个关闭按钮，并添加到了一个菜单中，设置了按钮的位置，让它在场景中显现，Metro程序并没有提供关闭程序的接口，所以点击无效。



COCOS2D X

helloworld :

设置文字，字体，字号

```
auto label = Label::createWithTTF("Hello World", "fonts/Marker Felt.ttf", 24);

// position the label on the center of the screen
label->setPosition(Vec2(origin.x + visibleSize.width/2,
                        origin.y + visibleSize.height - label->getContentSize().height));

// add the label as a child to this layer
this->addChild(label, 1);

// add "HelloWorld" splash screen"
auto sprite = Sprite::create("HelloWorld.png");

// position the sprite on the center of the screen
sprite->setPosition(Vec2(visibleSize.width/2 + origin.x, visibleSize.height/2 + origin.y));

// add the sprite as a child to this layer
this->addChild(sprite, 0);

return true;
```

在这段代码中，创建了一个“Hello World”的label和一个精灵，这个精灵作为游戏的背景。



COCOS2D X

作业：

- 1、安装cocos2d-x
- 2、制作自己的hello world界面（要求界面中有自己的学号和姓名）
- 3、作业提交：提交实验报告（文档），Classes（文件夹），Resources（文件夹）。实验报告要求有截图。



THE END

THANKS FOR WATCHING

