## 如何创建层？

|  |
| --- |
|  |

**注意，很多时候都可以使用层的切换来代替场景的切换，这样子效率更高，层只需要加载一次，其他时间就可以根据需要显示或者隐藏对应的层**

**不过有时候切换场景是必须的。如从登录界面切换到游戏主场景。**

**层的默认锚点是在左下角，这个跟很多UI组件的锚点在正中心是不一样的**

## 实例，层的简单使用

|  |  |
| --- | --- |
| **//MyLayer.h**  #pragma once  #include "cocos2d.h"  USING\_NS\_CC;  class MyLayer : public Layer  {  public:    virtual bool init();  //重写触摸事件  virtual bool onTouchBegan(Touch\* touch,Event\* event );  virtual void onTouchMoved(Touch\* touch, Event\* event);  virtual void onTouchCancelled(Touch\* touch, Event\* event);  virtual void onTouchEnded(Touch\* touch, Event\* event);  // a selector callback  // void menuCloseCallback(cocos2d::Ref\* pSender);    // implement the "static create()" method manually  CREATE\_FUNC(MyLayer);  private:    }; | **//MyLayer.cpp**  #include "MyLayer.h"  bool MyLayer::init()  {  if(!Layer::init())  return false;  //设置层的大小  this->setContentSize(Director::getInstance()->getVisibleSize());  //设置单点模式  this->setTouchMode(Touch::DispatchMode::ONE\_BY\_ONE);  //启用触摸模式  this->setTouchEnabled(true);//这里如果是false的话，触摸事件会传递给下一层  return true;  }  bool MyLayer::onTouchBegan(Touch \* touch, Event \* event)  {  CCLOG("Touch began....");  return true; //注意：如果这个方法返回false，后面的onTouchXXX方法都不会执行  }  void MyLayer::onTouchMoved(Touch \* touch, Event \* event)  {  CCLOG("Touch Moved....");  }  void MyLayer::onTouchCancelled(Touch \* touch, Event \* event)  {  CCLOG("Touch Cancelled....");  }  void MyLayer::onTouchEnded(Touch \* touch, Event \* event)  {  CCLOG("Touch Ended....");  } |
| **//HelloWorldScene.h**  #ifndef \_\_HELLOWORLD\_SCENE\_H\_\_  #define \_\_HELLOWORLD\_SCENE\_H\_\_  #include "cocos2d.h"  class HelloWorld : public cocos2d::Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();    // a selector callback  void menuCloseCallback(cocos2d::Ref\* pSender);    // implement the "static create()" method manually  CREATE\_FUNC(HelloWorld);  private:  //添加一个点击计数器  int m\_counter;  };  #endif // \_\_HELLOWORLD\_SCENE\_H\_\_ | **//HelloWorldScene.cpp**  #include "HelloWorldScene.h"  #include "SpriteScene2.h"  #include "SimpleAudioEngine.h"  #include "MyLayer.h"  USING\_NS\_CC;  Scene\* HelloWorld::createScene()  {  return HelloWorld::create();  }  // Print useful error message instead of segfaulting when files are not there.  static void problemLoading(const char\* filename)  {  printf("Error while loading: %s\n", filename);  printf("Depending on how you compiled you might have to add 'Resources/' in front of filenames in HelloWorldScene.cpp\n");  }  // on "init" you need to initialize your instance  bool HelloWorld::init()  {  //////////////////////////////  // 1. super init first  if ( !Scene::init() )  {  return false;  }  auto visibleSize = Director::getInstance()->getVisibleSize();  Vec2 origin = Director::getInstance()->getVisibleOrigin();  /////////////////////////////  // 2. add a menu item with "X" image, which is clicked to quit the program  // you may modify it.  // 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));  if (closeItem == nullptr ||  closeItem->getContentSize().width <= 0 ||  closeItem->getContentSize().height <= 0)  {  problemLoading("'CloseNormal.png' and 'CloseSelected.png'");  }  else  {  float x = origin.x + visibleSize.width - closeItem->getContentSize().width/2;  float y = origin.y + closeItem->getContentSize().height/2;  closeItem->setPosition(Vec2(x,y));  }  // create menu, it's an autorelease object  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  /////////////////////////////  // 3. add your codes below...  // add a label shows "Hello World"  // create and initialize a label  auto label = Label::createWithTTF("Hello World", "fonts/Marker Felt.ttf", 24);  if (label == nullptr)  {  problemLoading("'fonts/Marker Felt.ttf'");  }  else  {  // position the label on the center of the screen  label->setPosition(Vec2(origin.x + visibleSize.width/2,  origin.y + visibleSize.height - label->getContentSize().height));  /\*label->setPosition(Vec2(origin.x + visibleSize.width\*5 / 12,  origin.y + visibleSize.height - label->getContentSize().height));\*/  // add the label as a child to this layer  this->addChild(label, 1);  }  this->addChild(MyLayer::create());  // add "HelloWorld" splash screen"  auto sprite = Sprite::create("HelloWorld.png");  if (sprite == nullptr)  {  problemLoading("'HelloWorld.png'");  }  else  {  // 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);  }  //设置文本和图片左对齐  //写法1.  /\*label->setPosition(Vec2(origin.x + sprite->getContentSize().width,  origin.y + visibleSize.height - label->getContentSize().height));\*/  //写法2  label->setPositionX(label->getPosition().x - sprite->getContentSize().width / 2);  label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_BOTTOM\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_TOP\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_BOTTOM\_RIGHT);  //label->setAnchorPoint(Vec2::ANCHOR\_TOP\_RIGHT);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_TOP);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_BOTTOM);  m\_counter = 0;    return true;  }  void HelloWorld::menuCloseCallback(Ref\* pSender)  {  m\_counter++;  //Close the cocos2d-x game scene and quit the application  //Director::getInstance()->end();  //修改为切换场景,方法1  //Director::getInstance()->replaceScene(SpriteScene::createScene());  //修改为切换场景,方法2  if (m\_counter % 2 == 1) {  //Director::getInstance()->pushScene(TransitionFade::create(2.f, SpriteScene::createScene(), Color3B::RED));//添加切换场景效果  //Director::getInstance()->pushScene(TransitionSlideInB::create(2.f, SpriteScene::createScene()));//添加切换场景效果,从下往上滑动  //Director::getInstance()->pushScene(TransitionSlideInT::create(2.f, SpriteScene::createScene()));//添加切换场景效果。从上往下滑动  //Director::getInstance()->pushScene(TransitionSlideInL::create(2.f, SpriteScene::createScene()));//添加切换场景效果,从左往右  Director::getInstance()->pushScene(TransitionSlideInR::create(2.f, SpriteScene::createScene()));//添加切换场景效果，从右往左  //Director::getInstance()->pushScene(TransitionFlipX::create(2.f, SpriteScene::createScene()));//添加切换场景效果,翻转x轴  //Director::getInstance()->pushScene(TransitionFlipY::create(2.f, SpriteScene::createScene()));//添加切换场景效果，翻转y轴  }    #if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_IOS)  //exit(0);  #endif  /\*To navigate back to native iOS screen(if present) without quitting the application ,do not use Director::getInstance()->end() and exit(0) as given above,instead trigger a custom event created in RootViewController.mm as below\*/  //EventCustom customEndEvent("game\_scene\_close\_event");  //\_eventDispatcher->dispatchEvent(&customEndEvent);  } |
| **//AppDelegate.cpp**  #include "AppDelegate.h"  #include"SpriteScene2.h"  #include"HelloWorldScene.h"  // #define USE\_AUDIO\_ENGINE 1  // #define USE\_SIMPLE\_AUDIO\_ENGINE 1  #if USE\_AUDIO\_ENGINE && USE\_SIMPLE\_AUDIO\_ENGINE  #error "Don't use AudioEngine and SimpleAudioEngine at the same time. Please just select one in your game!"  #endif  #if USE\_AUDIO\_ENGINE  #include "audio/include/AudioEngine.h"  using namespace cocos2d::experimental;  #elif USE\_SIMPLE\_AUDIO\_ENGINE  #include "audio/include/SimpleAudioEngine.h"  using namespace CocosDenshion;  #endif  USING\_NS\_CC;  static cocos2d::Size designResolutionSize = cocos2d::Size(480, 320);  //static cocos2d::Size designResolutionSize = cocos2d::Size(600, 480);  static cocos2d::Size smallResolutionSize = cocos2d::Size(480, 320);  static cocos2d::Size mediumResolutionSize = cocos2d::Size(1024, 768);  static cocos2d::Size largeResolutionSize = cocos2d::Size(2048, 1536);  AppDelegate::AppDelegate()  {  }  AppDelegate::~AppDelegate()  {  #if USE\_AUDIO\_ENGINE  AudioEngine::end();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::end();  #endif  }  // if you want a different context, modify the value of glContextAttrs  // it will affect all platforms  void AppDelegate::initGLContextAttrs()  {  // set OpenGL context attributes: red,green,blue,alpha,depth,stencil  GLContextAttrs glContextAttrs = {8, 8, 8, 8, 24, 8};  GLView::setGLContextAttrs(glContextAttrs);  }  // if you want to use the package manager to install more packages,  // don't modify or remove this function  static int register\_all\_packages()  {  return 0; //flag for packages manager  }  bool AppDelegate::applicationDidFinishLaunching() {  // initialize director  auto director = Director::getInstance();  auto glview = director->getOpenGLView();  if(!glview) {  #if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_WIN32) || (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_MAC) || (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_LINUX)  glview = GLViewImpl::createWithRect("Hello", cocos2d::Rect(0, 0, designResolutionSize.width, designResolutionSize.height));  #else  glview = GLViewImpl::create("Hello");  #endif  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(1.0f / 60);  // Set the design resolution  glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::NO\_BORDER);  //glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::SHOW\_ALL);  auto frameSize = glview->getFrameSize();  // if the frame's height is larger than the height of medium size.  if (frameSize.height > mediumResolutionSize.height)  {  director->setContentScaleFactor(MIN(largeResolutionSize.height/designResolutionSize.height, largeResolutionSize.width/designResolutionSize.width));  }  // if the frame's height is larger than the height of small size.  else if (frameSize.height > smallResolutionSize.height)  {  director->setContentScaleFactor(MIN(mediumResolutionSize.height/designResolutionSize.height, mediumResolutionSize.width/designResolutionSize.width));  }  // if the frame's height is smaller than the height of medium size.  else  {  director->setContentScaleFactor(MIN(smallResolutionSize.height/designResolutionSize.height, smallResolutionSize.width/designResolutionSize.width));  }  register\_all\_packages();  // create a scene. it's an autorelease object  auto scene = HelloWorld::createScene();  //auto scene = SpriteScene::createScene();  // run  director->runWithScene(scene);  return true;  }  // This function will be called when the app is inactive. Note, when receiving a phone call it is invoked.  void AppDelegate::applicationDidEnterBackground() {  Director::getInstance()->stopAnimation();  #if USE\_AUDIO\_ENGINE  AudioEngine::pauseAll();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::getInstance()->pauseBackgroundMusic();  SimpleAudioEngine::getInstance()->pauseAllEffects();  #endif  }  // this function will be called when the app is active again  void AppDelegate::applicationWillEnterForeground() {  Director::getInstance()->startAnimation();  #if USE\_AUDIO\_ENGINE  AudioEngine::resumeAll();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::getInstance()->resumeBackgroundMusic();  SimpleAudioEngine::getInstance()->resumeAllEffects();  #endif  } |  |

**如果不设置颜色，层是透明的，被层覆盖的按钮还能够被点击。**

## 层的堆叠

|  |
| --- |
|  |

代码实例,这里只给出了Hello的代码，其他代码同上

|  |  |
| --- | --- |
| **//HelloWorldScene.h**  #ifndef \_\_HELLOWORLD\_SCENE\_H\_\_  #define \_\_HELLOWORLD\_SCENE\_H\_\_  #include "cocos2d.h"  class HelloWorld : public cocos2d::Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();    // a selector callback  void menuCloseCallback(cocos2d::Ref\* pSender);    // implement the "static create()" method manually  CREATE\_FUNC(HelloWorld);  private:  //添加一个点击计数器  int m\_counter;  };  #endif // \_\_HELLOWORLD\_SCENE\_H\_\_ | **//HelloWorldScene.cpp**  #include "HelloWorldScene.h"  #include "SpriteScene2.h"  #include "SimpleAudioEngine.h"  #include "MyLayer.h"  USING\_NS\_CC;  Scene\* HelloWorld::createScene()  {  return HelloWorld::create();  }  // Print useful error message instead of segfaulting when files are not there.  static void problemLoading(const char\* filename)  {  printf("Error while loading: %s\n", filename);  printf("Depending on how you compiled you might have to add 'Resources/' in front of filenames in HelloWorldScene.cpp\n");  }  // on "init" you need to initialize your instance  bool HelloWorld::init()  {  //////////////////////////////  // 1. super init first  if ( !Scene::init() )  {  return false;  }  auto visibleSize = Director::getInstance()->getVisibleSize();  Vec2 origin = Director::getInstance()->getVisibleOrigin();  /////////////////////////////  // 2. add a menu item with "X" image, which is clicked to quit the program  // you may modify it.  // 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));  if (closeItem == nullptr ||  closeItem->getContentSize().width <= 0 ||  closeItem->getContentSize().height <= 0)  {  problemLoading("'CloseNormal.png' and 'CloseSelected.png'");  }  else  {  float x = origin.x + visibleSize.width - closeItem->getContentSize().width/2;  float y = origin.y + closeItem->getContentSize().height/2;  closeItem->setPosition(Vec2(x,y));  }  // create menu, it's an autorelease object  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  /////////////////////////////  // 3. add your codes below...  // add a label shows "Hello World"  // create and initialize a label  auto label = Label::createWithTTF("Hello World", "fonts/Marker Felt.ttf", 24);  if (label == nullptr)  {  problemLoading("'fonts/Marker Felt.ttf'");  }  else  {  // position the label on the center of the screen  label->setPosition(Vec2(origin.x + visibleSize.width/2,  origin.y + visibleSize.height - label->getContentSize().height));  /\*label->setPosition(Vec2(origin.x + visibleSize.width\*5 / 12,  origin.y + visibleSize.height - label->getContentSize().height));\*/  // add the label as a child to this layer  this->addChild(label, 1);  }  //this->addChild(MyLayer::create());  // add "HelloWorld" splash screen"  auto sprite = Sprite::create("HelloWorld.png");  if (sprite == nullptr)  {  problemLoading("'HelloWorld.png'");  }  else  {  // 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);  }  //设置文本和图片左对齐  //写法1.  /\*label->setPosition(Vec2(origin.x + sprite->getContentSize().width,  origin.y + visibleSize.height - label->getContentSize().height));\*/  //写法2  label->setPositionX(label->getPosition().x - sprite->getContentSize().width / 2);  label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_BOTTOM\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_TOP\_LEFT);  //label->setAnchorPoint(Vec2::ANCHOR\_BOTTOM\_RIGHT);  //label->setAnchorPoint(Vec2::ANCHOR\_TOP\_RIGHT);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_TOP);  //label->setAnchorPoint(Vec2::ANCHOR\_MIDDLE\_BOTTOM);  m\_counter = 0;  //不使用自定义层，使用更简单的方式  //默认情况下后添加的层会覆盖先添加的层，不过可以使用setZOrder(n)方法来修改顺序  //创建红色层  LayerColor\* redLayer = LayerColor::create(Color4B::RED);  this->addChild(redLayer);  redLayer->setZOrder(10);  //创建绿色层  LayerColor\* greenLayer = LayerColor::create(Color4B::GREEN);  this->addChild(greenLayer);  return true;  }  void HelloWorld::menuCloseCallback(Ref\* pSender)  {  m\_counter++;  //Close the cocos2d-x game scene and quit the application  //Director::getInstance()->end();  //修改为切换场景,方法1  //Director::getInstance()->replaceScene(SpriteScene::createScene());  //修改为切换场景,方法2  if (m\_counter % 2 == 1) {  //Director::getInstance()->pushScene(TransitionFade::create(2.f, SpriteScene::createScene(), Color3B::RED));//添加切换场景效果  //Director::getInstance()->pushScene(TransitionSlideInB::create(2.f, SpriteScene::createScene()));//添加切换场景效果,从下往上滑动  //Director::getInstance()->pushScene(TransitionSlideInT::create(2.f, SpriteScene::createScene()));//添加切换场景效果。从上往下滑动  //Director::getInstance()->pushScene(TransitionSlideInL::create(2.f, SpriteScene::createScene()));//添加切换场景效果,从左往右  Director::getInstance()->pushScene(TransitionSlideInR::create(2.f, SpriteScene::createScene()));//添加切换场景效果，从右往左  //Director::getInstance()->pushScene(TransitionFlipX::create(2.f, SpriteScene::createScene()));//添加切换场景效果,翻转x轴  //Director::getInstance()->pushScene(TransitionFlipY::create(2.f, SpriteScene::createScene()));//添加切换场景效果，翻转y轴  }    #if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_IOS)  //exit(0);  #endif  /\*To navigate back to native iOS screen(if present) without quitting the application ,do not use Director::getInstance()->end() and exit(0) as given above,instead trigger a custom event created in RootViewController.mm as below\*/  //EventCustom customEndEvent("game\_scene\_close\_event");  //\_eventDispatcher->dispatchEvent(&customEndEvent);  } |

## 视差滚动

### 1.什么是视差滚动

|  |
| --- |
|  |

### 2. 视差滚动的规则

|  |
| --- |
|  |

|  |
| --- |
|  |

## 视差滚动示例

|  |
| --- |
|  |

**代码实现**

|  |  |
| --- | --- |
| **//ParallaxScene.h**  #pragma once  #include"cocos2d.h"  #include"ui/CocosGUI.h"  USING\_NS\_CC;  using namespace ui;  class ParallaxScene :public cocos2d::Scene {  public:    virtual bool init();  // a selector callback  void menuCloseCallback(cocos2d::Ref\* pSender);  // implement the "static create()" method manually  CREATE\_FUNC(ParallaxScene);  private:  ParallaxNode\* m\_node;  float viewSpeed;  float contentSpeed;  float texSpeed;  float bgSpeed;  //视图层最大移动距离  float m\_distance;  }; | **//ParallaxScene.cpp**  #include "ParallaxScene.h"  bool ParallaxScene::init()  {  if(!Scene::init())  return false;  //获取屏幕大小  auto visibleSize = Director::getInstance()->getVisibleSize();  //创建背景层,设置长度最短  auto bgLayer = LayerColor::create(Color4B(0,127,213,255));  bgLayer->setContentSize(Size(1280,visibleSize.height));  //层的默认锚点是在左下角    auto sprite = Sprite::create("parallaxNode/cloud3.png");  sprite->setPosition(Vec2(bgLayer->getContentSize()/2));  //sprite->setScale(0.25f);  bgLayer->addChild(sprite);  //创建贴图层设置长度次短  auto texLayer = Layer::create();  texLayer->setContentSize(Size(1440, visibleSize.height));  auto texSprite = Sprite::create("parallaxNode/cloud5.png");  texSprite->setScale(0.5f);  texSprite->setPosition(Vec2(texLayer->getContentSize().width \* 0.75, texLayer->getContentSize() .height/ 2));  texLayer->addChild(texSprite);  //创建内容层设置长度最长  auto contentLayer = Layer::create();  contentLayer->setContentSize(Size(1920, visibleSize.height));  auto contentSprite = Sprite::create("parallaxNode/cloud4.png");  contentSprite->setScale(0.8f);  contentSprite->setPosition(Vec2(contentLayer->getContentSize().width \* 0.4, contentLayer->getContentSize() .height/ 2));  contentLayer->addChild(contentSprite);  //计算各个层的速度  viewSpeed = 1.f;  //计算内容层的速度  contentSpeed = viewSpeed;  //计算贴图层的速度  texSpeed = contentSpeed \* (texLayer->getContentSize().width - visibleSize.width) / (contentLayer->getContentSize().width - visibleSize.width);  //计算背景层的速度  bgSpeed = contentSpeed\*(bgLayer->getContentSize().width - visibleSize.width) / (contentLayer->getContentSize().width - visibleSize.width);  //创建视图层节点  m\_node = ParallaxNode::create();  m\_node->addChild(bgLayer,1,Vec2(bgSpeed,1),Vec2::ZERO);  m\_node->addChild(texLayer,2, Vec2(texSpeed, 1), Vec2::ZERO);  m\_node->addChild(contentLayer,3,Vec2(contentSpeed, 1), Vec2::ZERO);  this->addChild(m\_node);  m\_distance = ((bgLayer->getContentSize().width - visibleSize.width) / bgSpeed) \* viewSpeed;    this->schedule([&](float dlt) {  if (m\_node->getPositionX() < -m\_distance) {  this->unschedule("schedule");  }  m\_node->setPositionX(m\_node->getPositionX() - viewSpeed);  },0.02f,"schedule");  return true;  }  void ParallaxScene::menuCloseCallback(cocos2d::Ref \* pSender)  {  } |
| **//AppDelegate.cpp**  #include "AppDelegate.h"  #include"SpriteScene2.h"  #include"HelloWorldScene.h"  #include"ParallaxScene.h"  // #define USE\_AUDIO\_ENGINE 1  // #define USE\_SIMPLE\_AUDIO\_ENGINE 1  #if USE\_AUDIO\_ENGINE && USE\_SIMPLE\_AUDIO\_ENGINE  #error "Don't use AudioEngine and SimpleAudioEngine at the same time. Please just select one in your game!"  #endif  #if USE\_AUDIO\_ENGINE  #include "audio/include/AudioEngine.h"  using namespace cocos2d::experimental;  #elif USE\_SIMPLE\_AUDIO\_ENGINE  #include "audio/include/SimpleAudioEngine.h"  using namespace CocosDenshion;  #endif  USING\_NS\_CC;  //static cocos2d::Size designResolutionSize = cocos2d::Size(480, 320);  static cocos2d::Size designResolutionSize = cocos2d::Size(960, 540);  static cocos2d::Size smallResolutionSize = cocos2d::Size(480, 320);  static cocos2d::Size mediumResolutionSize = cocos2d::Size(1024, 768);  static cocos2d::Size largeResolutionSize = cocos2d::Size(2048, 1536);  AppDelegate::AppDelegate()  {  }  AppDelegate::~AppDelegate()  {  #if USE\_AUDIO\_ENGINE  AudioEngine::end();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::end();  #endif  }  // if you want a different context, modify the value of glContextAttrs  // it will affect all platforms  void AppDelegate::initGLContextAttrs()  {  // set OpenGL context attributes: red,green,blue,alpha,depth,stencil  GLContextAttrs glContextAttrs = {8, 8, 8, 8, 24, 8};  GLView::setGLContextAttrs(glContextAttrs);  }  // if you want to use the package manager to install more packages,  // don't modify or remove this function  static int register\_all\_packages()  {  return 0; //flag for packages manager  }  bool AppDelegate::applicationDidFinishLaunching() {  // initialize director  auto director = Director::getInstance();  auto glview = director->getOpenGLView();  if(!glview) {  #if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_WIN32) || (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_MAC) || (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_LINUX)  glview = GLViewImpl::createWithRect("Hello", cocos2d::Rect(0, 0, designResolutionSize.width, designResolutionSize.height));  #else  glview = GLViewImpl::create("Hello");  #endif  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(1.0f / 60);  // Set the design resolution  glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::NO\_BORDER);  //glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::SHOW\_ALL);  auto frameSize = glview->getFrameSize();  // if the frame's height is larger than the height of medium size.  if (frameSize.height > mediumResolutionSize.height)  {  director->setContentScaleFactor(MIN(largeResolutionSize.height/designResolutionSize.height, largeResolutionSize.width/designResolutionSize.width));  }  // if the frame's height is larger than the height of small size.  else if (frameSize.height > smallResolutionSize.height)  {  director->setContentScaleFactor(MIN(mediumResolutionSize.height/designResolutionSize.height, mediumResolutionSize.width/designResolutionSize.width));  }  // if the frame's height is smaller than the height of medium size.  else  {  director->setContentScaleFactor(MIN(smallResolutionSize.height/designResolutionSize.height, smallResolutionSize.width/designResolutionSize.width));  }  register\_all\_packages();  // create a scene. it's an autorelease object  //auto scene = HelloWorld::createScene();  //auto scene = SpriteScene::createScene();  auto scene = ParallaxScene::create();  // run  director->runWithScene(scene);  return true;  }  // This function will be called when the app is inactive. Note, when receiving a phone call it is invoked.  void AppDelegate::applicationDidEnterBackground() {  Director::getInstance()->stopAnimation();  #if USE\_AUDIO\_ENGINE  AudioEngine::pauseAll();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::getInstance()->pauseBackgroundMusic();  SimpleAudioEngine::getInstance()->pauseAllEffects();  #endif  }  // this function will be called when the app is active again  void AppDelegate::applicationWillEnterForeground() {  Director::getInstance()->startAnimation();  #if USE\_AUDIO\_ENGINE  AudioEngine::resumeAll();  #elif USE\_SIMPLE\_AUDIO\_ENGINE  SimpleAudioEngine::getInstance()->resumeBackgroundMusic();  SimpleAudioEngine::getInstance()->resumeAllEffects();  #endif  } |  |