## 什么是九宫格?

|  |
| --- |
|  |

九宫格的长宽比例是可以改变的。九宫格是中心区域是非常重要的，需要设置。九宫格的四个角是**不会**随着中心区域的变的改变而改变

可以将中心区域的图片做得非常小，可以是一个像素。

Scale9Sprite

创建九宫格精灵

|  |
| --- |
|  |

## 九宫格精灵改变大小可以用setContentSize(Size(x,y))这个方法，因为这个方法可以改变形状

当然，修改形状也可以分别调用setScaleX(n)和setScaleY(m)来设置，m！=n就 行了，不过上面的方法简单一些。

九宫格精灵实例

|  |  |
| --- | --- |
| //Scale9SpriteScene.h  #pragma once  #include"cocos2d.h"  #include"ui/UIScale9Sprite.h"  USING\_NS\_CC;  using namespace ui;  #include<string>  using namespace std;  class Scale9SpriteScene:public Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();  //menu callback  void menuCloseCallback(cocos2d::Ref\* sender);  CREATE\_FUNC(Scale9SpriteScene);  private:    }; | //Scale9SpriteScene.cpp  #include "Scale9SpriteScene.h"  cocos2d::Scene \* Scale9SpriteScene::createScene()  {  return Scale9SpriteScene::create();  }  bool Scale9SpriteScene::init()  {  if (!Scene::init()) return false;  //获取可视区域大小  auto visibleSize = Director::getInstance()->getVisibleSize();  auto origin = Director::getInstance()->getVisibleOrigin();  //获取原点坐标  //创建关闭按钮  auto closeItem = MenuItemImage::create("CloseNormal.png", "CloseSelected.png", CC\_CALLBACK\_1(Scale9SpriteScene::menuCloseCallback, this));  //设置位置  float x = origin.x + visibleSize.width - closeItem->getContentSize().width / 2;  float y = origin.y + closeItem->getContentSize().height / 2;  closeItem->setPosition(Vec2(x, y));  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  //添加一个白色背景  //auto layer = LayerColor::create(Color4B(255,255,255,255));  auto layer = LayerColor::create(Color4B::WHITE);  this->addChild(layer);  //创建九宫格精灵  auto sprite = Scale9Sprite::create("scale9sprite/button9.png");  //设置九宫格中心  sprite->setCapInsets(Rect(4,4,22,22));  this->addChild(sprite);  //精灵的锚点默认在正中心，这里可以不用设置  sprite->setPosition(Vec2(visibleSize / 2));  //sprite->setScaleX(100);//只对x轴方向进行缩放  //sprite->setScaleY(30);//只对y轴方向进行缩放  sprite->setContentSize(Size(100, 100));//这个方法的修改相当于上面两句的效果    return true;  }  void Scale9SpriteScene::menuCloseCallback(cocos2d::Ref \* sender)  {  Director::getInstance()->end();  } |
| //AppDelegate.cpp  #include "AppDelegate.h"  //#include"SpriteScene.h"  #include"Scale9SpriteScene.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(755, 544);  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 = Scale9SpriteScene::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  } |  |

九宫格精灵vs普通精灵

|  |
| --- |
|  |

九宫格精灵的应用范围：

|  |
| --- |
|  |

演示实例

|  |
| --- |
|  |

|  |  |
| --- | --- |
| //Scale9SpriteScene.h  #pragma once  #include"cocos2d.h"  #include"ui/UIScale9Sprite.h"  USING\_NS\_CC;  using namespace ui;  #include<string>  using namespace std;  class Scale9SpriteScene:public Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();  //menu callback  void menuCloseCallback(cocos2d::Ref\* sender);  CREATE\_FUNC(Scale9SpriteScene);  private:    }; | //Scale9SpriteScene.cpp  #include "Scale9SpriteScene.h"  cocos2d::Scene \* Scale9SpriteScene::createScene()  {  return Scale9SpriteScene::create();  }  bool Scale9SpriteScene::init()  {  if (!Scene::init()) return false;  //获取可视区域大小  auto visibleSize = Director::getInstance()->getVisibleSize();  auto origin = Director::getInstance()->getVisibleOrigin();  //获取原点坐标  //创建关闭按钮  auto closeItem = MenuItemImage::create("CloseNormal.png", "CloseSelected.png", CC\_CALLBACK\_1(Scale9SpriteScene::menuCloseCallback, this));  //设置位置  float x = origin.x + visibleSize.width - closeItem->getContentSize().width / 2;  float y = origin.y + closeItem->getContentSize().height / 2;  closeItem->setPosition(Vec2(x, y));  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  //添加一个白色背景  //auto layer = LayerColor::create(Color4B(255,255,255,255));  auto layer = LayerColor::create(Color4B::WHITE);  this->addChild(layer);  //创建九宫格精灵  //auto sprite = Scale9Sprite::create("scale9sprite/square.png");  auto sprite = Scale9Sprite::create("scale9sprite/button9.png");  //设置九宫格中心  sprite->setCapInsets(Rect(20,20,1,1));  this->addChild(sprite);  //精灵的锚点默认在正中心，这里可以不用设置  sprite->setPosition(Vec2(visibleSize / 2));  //sprite->setScaleX(100);//只对x轴方向进行缩放  //sprite->setScaleY(30);//只对y轴方向进行缩放  //sprite->setContentSize(Size(100, 100));//这个方法的修改相当于上面两句的效果  //加载中文  auto dict = Dictionary::createWithContentsOfFile("tips.xml");  auto str = ((String\*)dict->objectForKey("bgkstory"))->getCString();  auto str2 = ((String\*)dict->objectForKey("lesson"))->getCString();  //log(str);//中文加载成功  //创建Label  auto label = Label::create(str2, "Arial", 24);  sprite->addChild(label);  //设置label的锚点  label->setAnchorPoint(Vec2::ANCHOR\_BOTTOM\_LEFT);//设置锚点在左下角  label->setPosition(Vec2(20,20));  //设置label文字显示宽度  if (label->getContentSize().width > 300)  {  //换行处理  label->setDimensions(300,0);//只是设置宽度，高度不变，所以这里是0  }  sprite->setContentSize(label->getContentSize()+Size(40,40));  return true;  }  void Scale9SpriteScene::menuCloseCallback(cocos2d::Ref \* sender)  {  Director::getInstance()->end();  } |
| //AppDelegate.cpp  #include "AppDelegate.h"  //#include"SpriteScene.h"  #include"Scale9SpriteScene.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 = Scale9SpriteScene::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  } | 效果 |