## 综合案例：翻卡牌

|  |
| --- |
|  |

## 案例代码实现

|  |  |
| --- | --- |
| //CardScene.h  #pragma once  #include"ui/UIButton.h"  #include "cocos2d.h"  //#include"ui/CocosGUI.h"  USING\_NS\_CC;  using namespace ui;  class CardScene : 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(CardScene);  private:  //卡牌  Button\* m\_btnCard;  //帧动画精灵  Sprite\* m\_sprite;  Animate\* m\_anim;  Animation\* m\_animation;  //卡牌出现  void showCard();  //点击卡牌翻转  void flipCard();  //帧动画播放  void playAnimation();  }; | //CardScene.cpp  #include "CardScene.h"  USING\_NS\_CC;  using namespace ui;  bool CardScene::init()  {  if (!Scene::init())return false;  //获取屏幕大小  auto visibleSize = Director::getInstance()->getVisibleSize();  //获取原点坐标  auto origin = Director::getInstance()->getVisibleOrigin();  //设置起始坐标  Vec2 position = Vec2(visibleSize / 2);  //创建关闭按钮  auto closeItem = MenuItemImage::create("CloseNormal.png", "CloseSelected.png", CC\_CALLBACK\_1(CardScene::menuCloseCallback, this));  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);  //翻卡牌动画  //添加一个背景  Sprite\* spr = Sprite::create("knifetower/back.jpg");  this->addChild(spr,1);  spr->setPosition(Vec2(visibleSize / 2));  spr->setContentSize(visibleSize);  //创建卡牌  m\_btnCard = Button::create("knifetower/card.png", "knifetower/card.png", "knifetower/card.png");  this->addChild(m\_btnCard,2);  m\_btnCard->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  m\_btnCard->setPosition(Vec2(visibleSize / 2));  m\_btnCard->setScale(0.2f);  //添加按钮点击事件  m\_btnCard->addClickEventListener([this](Ref\* sender) {  auto btn = static\_cast<Button\*>(sender);  btn->setEnabled(false);//点击后就不能再点击  //调用flipCard方法  flipCard();  //playAnimation();  });  //卡牌显示动画  showCard();    return true;  }  void CardScene::menuCloseCallback(cocos2d::Ref \* pSender)  {  Director::getInstance()->end();  }  void CardScene::showCard()  {  //旋转同时放大卡牌  auto rotAction = RotateTo::create(0.5f,2160.f);  auto sclAction = ScaleTo::create(0.5, .8f);  auto spawn = Spawn::createWithTwoActions(rotAction, sclAction);  m\_btnCard->runAction(spawn);  }  void CardScene::flipCard()  {  //1,将卡背x轴缩小到0，  auto sclXAction = ScaleTo::create(0.5f, 0,0.5);  //2.播放帧动画  auto callback = CallFunc::create([this]() {  //转换到卡面  m\_btnCard->loadTextures("knifetower/back2.png", "knifetower/back2.png", "knifetower/back2.png");  //播放动画  playAnimation();  });  //将背景恢复回来  auto sclX2Action = ScaleTo::create(0.5f, 0.8f, 0.8f);  m\_btnCard->runAction(Sequence::create(sclXAction,callback, sclX2Action,nullptr));  }  void CardScene::playAnimation()  {  //创建帧动画  auto cache = SpriteFrameCache::getInstance();  cache->addSpriteFramesWithFile("knifetower/kisses.plist");  m\_sprite = Sprite::createWithSpriteFrameName("1");  Vector<SpriteFrame\*> frames;  for (int i = 1; i <= 6; i++) {  frames.pushBack(cache->getSpriteFrameByName(Value(i).asString()));  }  m\_animation = Animation::createWithSpriteFrames(frames,1.f/frames.size());  m\_anim = Animate::create(m\_animation);    m\_sprite->setPosition(Vec2(m\_btnCard->getContentSize()/2)+Vec2(0,10));  m\_btnCard->addChild(m\_sprite);  //设置按钮在播放动画的时候不能点击  //m\_btnCard->setEnabled(false);  m\_sprite->runAction(RepeatForever::create(m\_anim));  } |
| //AppDelegate.cpp  #include "AppDelegate.h"  //#include"AnimateScene.h"  #include "CardScene.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, 400);  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 = ActionScene::create();  //auto scene = AnimateScene::create();  auto scene = CardScene::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  } |  |