1.文本框介绍

|  |
| --- |
|  |

1.基本使用例子

|  |  |
| --- | --- |
| //TextFieldScene.h  #pragma once  #include"cocos2d.h"  #include"ui/CocosGUI.h"  USING\_NS\_CC;  using namespace ui;  #include<string>  using namespace std;  class TextFieldScene:public Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();  //menu callback  void menuCloseCallback(cocos2d::Ref\* sender);  CREATE\_FUNC(TextFieldScene);  private:    }; | // TextFieldScene.cpp  #include "TextFieldScene.h"  using namespace ui;  #include<string>  using namespace std;  cocos2d::Scene \* TextFieldScene::createScene()  {  return TextFieldScene::create();  }  bool TextFieldScene::init()  {  if (!Scene::init()) return false;  //获取可视区域大小  auto visibleSize = Director::getInstance()->getVisibleSize();  //获取原点坐标  auto orgin = Director::getInstance()->getVisibleOrigin();  //创建关闭按钮  auto closeItem = MenuItemImage::create("CloseNormal.png", "CloseSelected.png", CC\_CALLBACK\_1(TextFieldScene::menuCloseCallback, this));  float x = orgin.x + visibleSize.width - closeItem->getContentSize().width / 2;  float y = orgin.y + closeItem->getContentSize().height / 2;  closeItem->setPosition(Vec2(x, y));  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  //创建文本框  TextField\* input = TextField::create();  //设置位置  input->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  input->setPosition(Vec2(visibleSize / 2));  //设置文本框属性  input->setFontName("Arial");  input->setFontSize(36);  //设置颜色  //input->setColor(Color3B::WHITE);  input->setTextColor(Color4B::WHITE);  //设置占位文字  input->setPlaceHolder("Please Enter Text Here");  //设置自动换行  input->ignoreContentAdaptWithSize(false);  //设置文本区域大小  input->setTextAreaSize(Size(300, 150));  //限制输入的最大长度  //input->setMaxLengthEnabled(true);//先启用最大长度限制功能然后设置最大输入长度才有用  //input->setMaxLength(3);//注意这里设置为3后可以输入3个数字，字母和中文字，这个和按字节计算不一样！！！  //添加文本框点击事件  input->addClickEventListener([&](Ref\* sender) { //输入文本后点击文本消失，但是placeholder的内容不会消失  dynamic\_cast<TextField\*>(sender)->setString("");    });  //添加文本改变事件  input->addEventListener([&](Ref\* sender, TextField::EventType type) {  //当用户输入中文是提示输入英文和数字  auto txtf = dynamic\_cast<TextField\*>(sender);  if (type == TextField::EventType::INSERT\_TEXT) {  string content = txtf->getString();  for (char c : content) {  if (c <= 0 || c >= 127) {  txtf->setString("");  log("please enter numbers and letters...");  break;  }  }  }  });  this->addChild(input);  return true;  }  void TextFieldScene::menuCloseCallback(cocos2d::Ref \* sender)  {  Director::getInstance()->end();  } |
| //AppDelegate.cpp  #include "AppDelegate.h"  #include"TextFieldScene.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 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);  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 = TextFieldScene::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  } |  |

2.实战例子



需要的资源

|  |
| --- |
|  |

|  |  |
| --- | --- |
| //TextFieldScene.h  #pragma once  #include"cocos2d.h"  #include"ui/CocosGUI.h"  USING\_NS\_CC;  using namespace ui;  #include<string>  using namespace std;  class TextFieldScene:public Scene  {  public:  static cocos2d::Scene\* createScene();  virtual bool init();  //menu callback  void menuCloseCallback(cocos2d::Ref\* sender);  CREATE\_FUNC(TextFieldScene);  private:    }; | // TextFieldScene.cpp  #include "TextFieldScene.h"  using namespace ui;  #include<string>  using namespace std;  cocos2d::Scene \* TextFieldScene::createScene()  {  return TextFieldScene::create();  }  bool TextFieldScene::init()  {  if (!Scene::init()) return false;  //获取可视区域大小  auto visibleSize = Director::getInstance()->getVisibleSize();  //获取原点坐标  auto orgin = Director::getInstance()->getVisibleOrigin();  //创建关闭按钮  auto closeItem = MenuItemImage::create("CloseNormal.png", "CloseSelected.png", CC\_CALLBACK\_1(TextFieldScene::menuCloseCallback, this));  float x = orgin.x + visibleSize.width - closeItem->getContentSize().width / 2;  float y = orgin.y + closeItem->getContentSize().height / 2;  closeItem->setPosition(Vec2(x, y));  auto menu = Menu::create(closeItem, NULL);  menu->setPosition(Vec2::ZERO);  this->addChild(menu, 1);  //添加一个背景图片  Sprite\* bg = Sprite::create("textfield/medbg.png");  bg->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  bg->setPosition(Vec2(visibleSize / 2));  this->addChild(bg);  //创建文本框  TextField\* input = TextField::create();  //设置位置  input->setAnchorPoint(Vec2::ANCHOR\_MIDDLE);  input->setPosition(Vec2(visibleSize / 2)+Vec2(19,-17));  //设置文本框属性  input->setFontName("Arial");  input->setFontSize(18);  //设置颜色  //input->setColor(Color3B::WHITE);  input->setTextColor(Color4B::WHITE);  //设置占位文字  //input->setPlaceHolder("Please Enter Text Here");  input->setString("Super Fighter");  //设置自动换行  input->ignoreContentAdaptWithSize(false);  //设置文本区域大小  input->setTextAreaSize(Size(200, 35));  //限制输入的最大长度  //input->setMaxLengthEnabled(true);//先启用最大长度限制功能然后设置最大输入长度才有用  //input->setMaxLength(3);//注意这里设置为3后可以输入3个数字，字母和中文字，这个和按字节计算不一样！！！  //添加文本框点击事件  input->addClickEventListener([&](Ref\* sender) { //输入文本后点击文本消失，但是placeholder的内容不会消失  dynamic\_cast<TextField\*>(sender)->setString("");    });  //添加文本改变事件  input->addEventListener([&](Ref\* sender, TextField::EventType type) {  //当用户输入中文是提示输入英文和数字  auto txtf = dynamic\_cast<TextField\*>(sender);  if (type == TextField::EventType::INSERT\_TEXT) {  string content = txtf->getString();  for (char c : content) {  if (c <= 0 || c >= 127) {  txtf->setString("");  log("please enter numbers and letters...");  break;  }  }  }  });  this->addChild(input);  return true;  }  void TextFieldScene::menuCloseCallback(cocos2d::Ref \* sender)  {  Director::getInstance()->end();  } |
| // AppDelegate.cpp  #include "AppDelegate.h"  #include"TextFieldScene.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();  auto scene = TextFieldScene::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  } |  |

**注意：cocos2dx的文本框没有形状，也没有光标在闪烁。如果想要有文本框的外观，需要添加背景精灵图片**