**现代操作系统应用开发实验报告**

——刘亚辉 16340157

Cocos2dx第九周实验报告

1. 参考资料
2. PPT课件 (导演，场景，层，精灵，菜单，标签)
3. <https://blog.csdn.net/sinat_28338727/article/details/51787882> (回调函数)
4. <http://www.cocoachina.com/bbs/read.php?tid=198291&keyword=cc%5C_CALLBACK> (回调函数参数问题)
5. 实验步骤

* 需求及流程：

1. 制作hello world界面，其中有自己的学号、姓名，更换背景图片
2. 设置文字样式、添加MenuItem，触发事件

* 详细实现过程：

（1）使用命令行创建HelloCocos项目：cocos new HelloCocos -l cpp -d D:\cocos

（2）更改应用的视图大小，设为mediumResolutionSize：

1. glview = GLViewImpl::createWithRect("HelloCocos", cocos2d::Rect(0, 0, mediumResolutionSize.width, mediumResolutionSize.height));

（3）设置label标签，显示学号：

1. // add a label shows "16340157"
2. // create and initialize a label
4. auto labelId = Label::createWithTTF("16340157", "fonts/Marker Felt.ttf", 24);
5. **if** (labelId == nullptr)
6. {
7. problemLoading("'fonts/Marker Felt.ttf'");
8. }
9. **else**
10. {
11. // position the label on the center of the screen
12. labelId->setPosition(Vec2(origin.x + visibleSize.width/2,
13. origin.y + visibleSize.height - 50));
14. labelId->setColor(Color3B(80,80,255));
15. //labelId->getContentSize().height
16. // add the label as a child to this layer
17. **this**->addChild(labelId, 1);
18. }

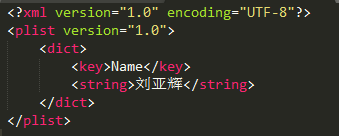
使用Label::createWithTTF方法创建学号的label，字体即为Marker Felt，并调用对象的方法来进行设置位置及颜色等，最后将其加入到场景中。

（4）之后设置姓名标签对象：

1. // add a label shows "刘亚辉"
2. // create and initialize a label
3. CCDictionary \*Name = CCDictionary::createWithContentsOfFile("Name.xml");
4. CCString\* myName = (CCString\*)Name->objectForKey("Name");
5. auto labelName = Label::createWithSystemFont(myName->getCString(), "华文中宋", 24);
6. **if** (labelName == nullptr)
7. {
8. problemLoading("'华文中宋'");
9. }
10. **else**
11. {
12. // position the label on the center of the screen
13. labelName->setPosition(Vec2(origin.x + visibleSize.width / 2,
14. origin.y + visibleSize.height - 50));
15. labelName->setColor(Color3B(80, 80, 255));
16. // add the label as a child to this layer
17. **this**->addChild(labelName, 1);
18. }
19. labelName->setVisible(**false**);

由于直接设置中文会出现乱码问题，则采用读取xml文件中键值对的方式获取中文姓名，然后在创建label至于场景中即可。

其中xml文件如下:



（5）设置响应事件，利用点击按钮，实现姓名与学号的交替显示：

1. auto toggleItem = MenuItemFont::create("Toggle", CC\_CALLBACK\_1(HelloWorld::menuToggleCallback, **this**, labelId, labelName));
2. **if** (toggleItem == nullptr)
3. {
4. problemLoading("'Toggle'");
5. }
6. **else**
7. {
8. **float** x = origin.x + visibleSize.width/2;
9. **float** y = origin.y + visibleSize.height - 100;
10. toggleItem->setPosition(Vec2(x, y));
11. }
12. // create menu, it's an autorelease object
13. menu = Menu::create(toggleItem, NULL);
14. menu->setPosition(Vec2::ZERO);
15. **this**->addChild(menu, 1);

创建一个文字按钮Toggle，设置回调函数，当点击效果触发时执行回调函数。设置的按钮需要创建一个Menu菜单对象，将菜单对象至于场景中即可。

回调函数menuToggleCallback如下（利用参数传递，将labelId和labelName传入回调函数）：

1. **void** HelloWorld::menuToggleCallback(Ref\* pSender, Label\* &labelId, Label\* &labelName)
2. {
3. //Close the cocos2d-x game scene and quit the application
4. **if** (labelId->isVisible()) {
5. labelId->setVisible(**false**);
6. labelName->setVisible(**true**);
7. }
8. **else** {
9. labelId->setVisible(**true**);
10. labelName->setVisible(**false**);
11. }
12. }

（6）创建背景图片：

1. // add "HelloWorld" splash screen"
2. auto sprite = Sprite::create("Background.jpg");
3. **if** (sprite == nullptr)
4. {
5. problemLoading("'Background.jpg'");
6. }
7. **else**
8. {
9. // position the sprite on the center of the screen
10. sprite->setPosition(Vec2(visibleSize.width/2 + origin.x, visibleSize.height/2 + origin.y));
12. // add the sprite as a child to this layer
13. **this**->addChild(sprite, 0);
14. }

背景图片需要创建为一个Sprite对象，设置为窗口的大小，并将其加入到场景中。细节，需要将背景图置于底层（0层），将其他对象置于1层。

1. 关键步骤截图

默认显示学号：



点击Toggle按钮之后：



1. 亮点与改进：

设置了MenuItem；设置中文文字样式；利用按钮回调函数来实现动态交互，事件响应。

1. 遇到的问题

回调函数参数问题，定义如下：

1. // new callbacks based on C++11
2. #define CC\_CALLBACK\_0(\_\_selector\_\_,\_\_target\_\_, ...) std::bind(&\_\_selector\_\_,\_\_target\_\_, ##\_\_VA\_ARGS\_\_)
3. #define CC\_CALLBACK\_1(\_\_selector\_\_,\_\_target\_\_, ...) std::bind(&\_\_selector\_\_,\_\_target\_\_, std::placeholders::\_1, ##\_\_VA\_ARGS\_\_)
4. #define CC\_CALLBACK\_2(\_\_selector\_\_,\_\_target\_\_, ...) std::bind(&\_\_selector\_\_,\_\_target\_\_, std::placeholders::\_1, std::placeholders::\_2, ##\_\_VA\_ARGS\_\_)
5. #define CC\_CALLBACK\_3(\_\_selector\_\_,\_\_target\_\_, ...) std::bind(&\_\_selector\_\_,\_\_target\_\_, std::placeholders::\_1, std::placeholders::\_2, std::placeholders::\_3, ##\_\_VA\_ARGS\_\_)

其中，\_\_selector\_\_是回调函数指针，\_\_target\_\_是回调对象指针，##\_\_VA\_ARGS\_\_是可变参数列表：CC\_CALLBACK\_0的this不是参数，而是执行这个回调函数的具体对象 然后，这个0也不能理解成多少个参数，而是指的占位符，比如某个回调函数有5个参数，如add（int，int，int，int，int）， 然后用CC\_CALLBACK\_2占上前两个参数，给后面三个参数绑定，可以写成 CC\_CALLBACK\_2(HelloCocos::add, this,15,15,15); 得到的func以后使用的时候直接给前两个参数赋值就ok了，如func（1， 2）。

1. 思考与总结