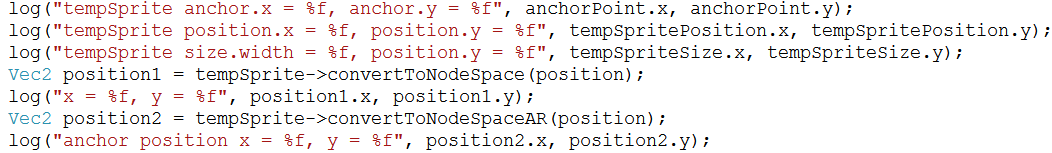
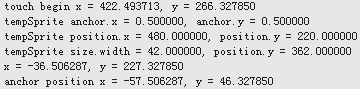
convertToNodeSpace与convertToNodeSpaceAR

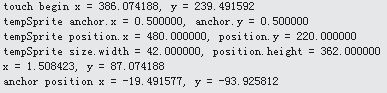
convertToNodeSpace是将屏幕坐标点转化为当前节点的局部坐标点（相对于左下角）

convertToNodeSpaceAR是将屏幕坐标点转化为当前节点的局部坐标点（相对于锚点）





但是如果我们将tempSprite顺时针旋转90度，即设置tempSprite->setRotation(90.0f);



仔细分析（注意：我们最后得到position结构都是基于局部坐标系）：



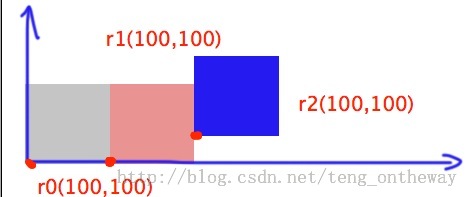
<http://blog.csdn.net/teng_ontheway/article/details/26078869>

游戏中经常会用到一些变换：

游戏中武器和角色在一个layer上，为了效率，会考虑将bullet, effect和 PhysicsParticle分别放到不用的层上，对应的层上使用batchnode来提高效率

武器和PhysicsParticleLauncher(粒子发射器)绑定，发射的时候，会向PhysicsParticleLayer的武器相同的位置上生成一个物理粒子特效

会经常用到convertToNodeSpace和convertToWorldSpace转换坐标



我们加三个sprite,r0,r1,r2,

r0的父节点是layer, 父节点下坐标(0,0,100,100)

r1的父节点是r0，父节点下坐标(0,0,100,100)

r2的父节点是r1，父节点下坐标(0,0,100,20)

**[cpp]** [view plain](http://blog.csdn.net/teng_ontheway/article/details/26078869) [copy](http://blog.csdn.net/teng_ontheway/article/details/26078869)

[在CODE上查看代码片](https://code.csdn.net/snippets/350426)

1. **bool** HelloWorld::init()
2. {
3. //////////////////////////////
4. // 1. super init first
5. **if** ( !Layer::init() ) {
6. **return** **false**;
7. }
9. // r0加入到根节点上(Layer) 坐标(0,0)
10. Sprite\* r0 = Sprite::create("data/pic/gun.png");
11. r0->setAnchorPoint(Point(0.f, 0.f));
12. **this**->addChild(r0);
14. // r1加入到r0上
15. Sprite\* r1 = Sprite::create("data/pic/gun.png");
16. r1->setAnchorPoint(Point(0.f, 0.f));
17. r1->setPosition(Point(100, 0));
18. r0->addChild(r1);
20. // r2加入到r1上
21. Sprite\* r2 = Sprite::create("data/pic/gun.png");
22. r2->setAnchorPoint(Point(0.f, 0.f));
23. r2->setPosition(Point(100, 20));
24. r1->addChild(r2);
26. }

**如何将r2的坐标转换为世界坐标？**

**如何将r2的坐标转换为r0的局部坐标？**

**[cpp]** [view plain](http://blog.csdn.net/teng_ontheway/article/details/26078869) [copy](http://blog.csdn.net/teng_ontheway/article/details/26078869)

[在CODE上查看代码片](https://code.csdn.net/snippets/350426)

1. // 获得r2的世界坐标，两种方法
2. // 方法1：以r2坐标系为起点，向根节点(世界坐标)变换，坐标必须为(0,0)
3. Point p2world = r2->convertToWorldSpace(Point(0, 0));
5. // 方法2：以r1坐标系为起点，向根节点(世界坐标)变换,坐标为r2在父节点r1内的坐标
6. p2world = r1->convertToWorldSpace(r2->getPosition());
8. // 获得r2在r0坐标系中的坐标
9. // convertToNodeSpace要传入的坐标为世界坐标
10. Point pt = r0->convertToNodeSpace(p2world);

方法如上：

工作原理详见cocos2dx代码

**[cpp]** [view plain](http://blog.csdn.net/teng_ontheway/article/details/26078869) [copy](http://blog.csdn.net/teng_ontheway/article/details/26078869)

[在CODE上查看代码片](https://code.csdn.net/snippets/350426)

1. Point Node::convertToWorldSpace(**const** Point& nodePoint) **const**
2. {
3. kmMat4 tmp = getNodeToWorldTransform();
4. kmVec3 vec3 = {nodePoint.x, nodePoint.y, 0};
5. kmVec3 ret;
6. kmVec3Transform(&ret, &vec3, &tmp);
7. **return** Point(ret.x, ret.y);
9. }
11. kmMat4 Node::getNodeToWorldTransform() **const**
12. {
13. kmMat4 t = **this**->getNodeToParentTransform();
15. **for** (Node \*p = \_parent; p != nullptr; p = p->getParent())
16. kmMat4Multiply(&t, &p->getNodeToParentTransform(), &t);
18. **return** t;
19. }