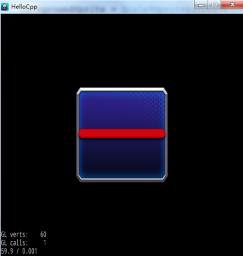
设置透明度可以使用setOpacity函数，如果父节点有串联关系，则优先获取父节点的透明度，但是注意，该函数同时也会更新Color值。同时如果设置了\_cascadeOpacityEnabled，则同时会更新子节点

对于Layer而言

//setCascadeOpacityEnabled(true);

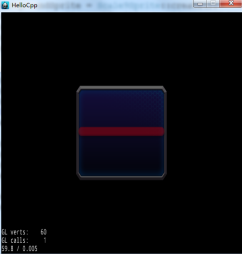
setOpacity(100);



变为：

setCascadeOpacityEnabled(true);

setOpacity(100);



以Sprite为例：

void Node::setOpacity(GLubyte opacity)

{

\_displayedOpacity = \_realOpacity = opacity;

**updateCascadeOpacity();**

}

**void Node::updateCascadeOpacity()**

{

GLubyte parentOpacity = 255;

**if (\_parent != nullptr && \_parent->isCascadeOpacityEnabled())**

**{**

**parentOpacity = \_parent->getDisplayedOpacity();**

**}**

**updateDisplayedOpacity(parentOpacity);**

}

**void Node::updateDisplayedOpacity(GLubyte parentOpacity)**

{

\_displayedOpacity = \_realOpacity \* parentOpacity/255.0;

**updateColor();**

if (\_cascadeOpacityEnabled)

{

for(const auto& child : \_children)

{

child->updateDisplayedOpacity(\_displayedOpacity);

}

}

}

**void Sprite::updateColor(void)**

{

Color4B color4( \_displayedColor.r, \_displayedColor.g, \_displayedColor.b, \_displayedOpacity );

// special opacity for premultiplied textures

if (\_opacityModifyRGB)

{

color4.r \*= \_displayedOpacity/255.0f;

color4.g \*= \_displayedOpacity/255.0f;

color4.b \*= \_displayedOpacity/255.0f;

}

for (ssize\_t i = 0; i < \_polyInfo.triangles.vertCount; i++) {

\_polyInfo.triangles.verts[i].colors = color4;

}

// renders using batch node

if (\_batchNode)

{

if (\_atlasIndex != INDEX\_NOT\_INITIALIZED)

{

\_textureAtlas->updateQuad(&\_quad, \_atlasIndex);

}

else

{

// no need to set it recursively

// update dirty\_, don't update recursiveDirty\_

setDirty(true);

}

}

// self render

// do nothing

}