UE4全中文小白入门系列视频教程，保证所有新人、小白可以快速上手UE4虚幻编辑器！

全套教程发布在bilibili网站上（大家可以直接点以下链接访问或在网站搜索教程标题）

1. 引擎获取及安装

<http://www.bilibili.com/video/av11241931/>

要点：UE4引擎的获取及编译安装

2. 简单场景搭建及蓝图应用小例子-感应灯

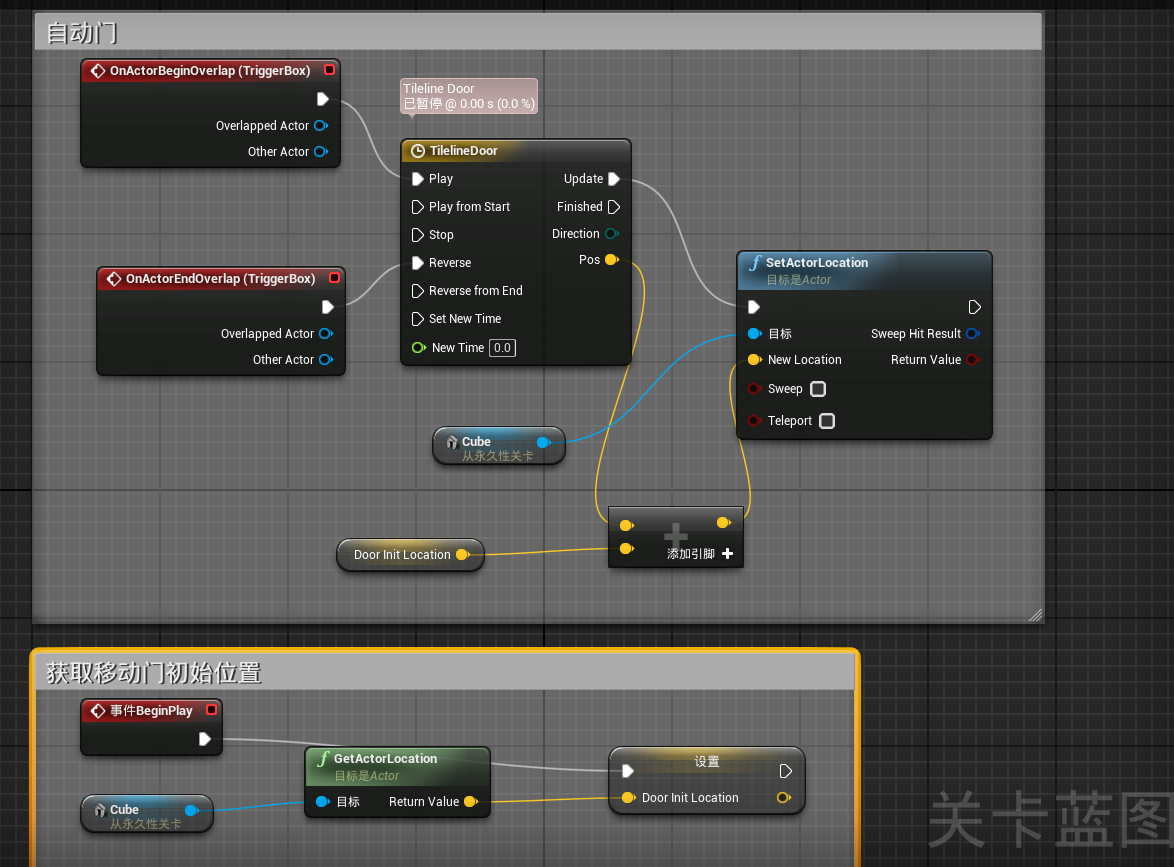
<http://www.bilibili.com/video/av11246194/>

要点：点光源，触发器，基本蓝图

3. 蓝图应用小例子-自动门

<http://www.bilibili.com/video/av11261040/>

要点：基本蓝图，时间轴动画



4. 利用蓝图类来复用对象

<http://www.bilibili.com/video/av11343833/>

要点：蓝图类

5.蓝图模拟台灯-用键盘实现无极调节光源亮度

<http://www.bilibili.com/video/av11468028/>

要点：蓝图键盘输入，定时器，自定义函数，流程控制节点，数学计算节点等

亮度值1000 -> 5000

For循环版本

{

Delay 0.1s

当前亮度 + 100

}

Timer定时器版本

间隔时间0.1s

0.1 s -> My Function

0.2s -> My Function

0.3s -> My Function

…

My Function {

判断当前亮度值<最大值

当前亮度+100

}

6.蓝图控制相机

<http://www.bilibili.com/video/av11563275/>

要点：蓝图控制相机，swich分支语句，定时器，部分内容在上一期教程中有讲解。

7.未完待续…

材质教程

**虚幻4材质全中文小白入门教程（一）- 介绍ue4材质系统**

1. 材质的概念，原理及作用

关键词：材质、纹理、贴图

材质：主要是描述模型表面的基础颜色、光照效果、质感、透明度等等，可以使用纹理(贴图)

纹理：材质表面使用的贴图，可以是一张图片

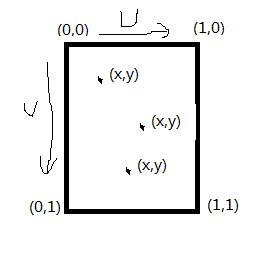
贴图：即一张图片

2. ue4材质编辑器的界面及基本操作

**虚幻4材质全中文小白入门教程（二）- 使用纹理及uv动画**

Uv: 2维坐标 是用来索引一张纹理的像素点位置

uv原点在纹理的左上角(0,0)



Uv动画：改变uv坐标 实现纹理的动画

**虚幻4材质全中文小白入门教程（三）- 简单平面水效果**

法线图+扰动uv模拟水面流动效果

法线图：记录每个顶点的法向量的一张图 vector3(x,y,z)，用于光照计算，影响模型表面的明暗变化

GLSL :

float timeFactor = 0.1;

float offsetFactor = 0.5;

vec3 normal = texture2D(u\_normalMap, v\_texCoord + vec2(CC\_Time.y \* timeFactor, 0)).xyz;

// simple calculate refraction UV offset

vec2 p = -1 + 2 \* v\_texCoord;

vec3 eyePos = vec3(0, 0, 10);

vec3 inVec = normalize(vec3(p, 0) - eyePos);

vec3 refractVec = refract(inVec, normal, 0.7);

vec2 v\_texCoordN = v\_texCoord;

v\_texCoordN += refractVec.xy \* offsetFactor;

gl\_FragColor = texture2D(CC\_Texture0, v\_texCoordN) ;

**虚幻4材质全中文小白入门教程（四）- 使用DepthFade深度消退实现水下透明度**

**要点：DepthFade节点的含义及使用**

**“深度消退”是在两个有穿插物体之间的一种透明度过渡，其中至少有一个物体是半透明材质**

**例如实现透过水面可以看到一部分水下的物体的效果**

**Depthfade 两个深度 1.场景深度 2.当前绘制的像素深度**

**虚幻4材质全中文小白入门教程（五）- 后期处理(Post Process)材质使用**

**BasePass->fog,sky->translucent->post process pass**

**1.Post Process Volume**

**进入体积区域的时候 会触发后效**

**2.Post Process Material**

**500 /1000 =0.5**

**100/1000 =0.1**

**1000/1000 =1**

**0~1000 /1000 = 0~1**

**0：黑色～ 1：白色**

**虚幻4流体模拟教程(一) –基于高度图交换RT**

**附上一个水池项目：**[**https://github.com/sitonmoon/FluidSimulationSurface**](https://github.com/sitonmoon/FluidSimulationSurface)

**原理：一种简单高效的波动算法。**

**技术细节：使用三张RT(渲染目标)交替，对上一帧形成的图像进行流体演算，得到下一帧图像。**

**准备两张RenderTarget：RT0， RT1 初始为空，使RT1作为每帧的显示（作为材质参数传进材质）**

**第1帧：**

**Drawcall0->由程序逻辑控制（如鼠标点击）在RT0上渲染一个圆形突起（以高度值形式记录）；**

**Drawcall 1->将RT0作为输入，进行流体演算，得到该突起圆进行波动后的图像，渲染到RT1，由于RT1作为显示，这时我们会看到水面第一次波动图像；**

**这时如果我们直接进入下一帧，由于下一帧会重复执行同样的流程，又把RT0作为输入，计算后渲染到RT1，这就变成重复一样的操作了，我们会看到水面一直是一个突起不变化，这当然不对，所以这里我们要增加一个步骤，即把RT0和RT1的指向进行交换。**

**学过基本编程的同学应该知道，两变量交换怎么做呢？**

**int a=1,b=2；要将a和b的值进行交换，不能直接a=b（这样就把a的值冲掉了），而需要借助一个临时变量c，先把a的值保存到c当中，再做交换：**

**int c=a; //临时保存a的值**

**a = b; //b赋予a**

**b = c; //完成交换**

**RT0和RT1的交换，需要借助另一个临时RT2，交换RT0和RT1的内容，这也就是开头我们需要三张RT的原因。这时我们可以进入下一帧了；**

**第2帧：由于每一帧执行的操作是完全一样的，这一帧重复之前的操作，但此时可能鼠标不进行点击，就是不需要再创建一个水面突起，直接进入Drawcall 1；**

**Drawcall 1->将RT0作为输入，注意此时RT0是上一次RT1的演算结果，进行流体演算得到第二次波动后的图像，渲染到RT1，看到水面第二次波动图像；**

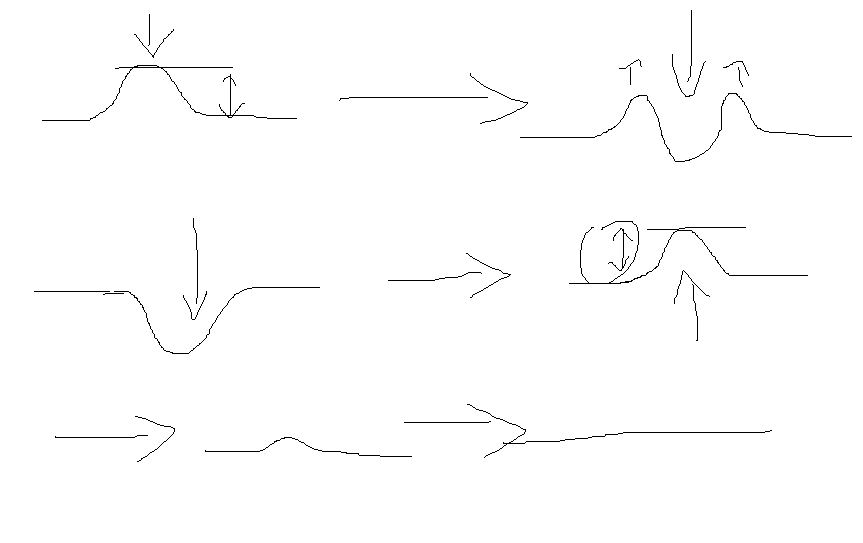
**交换RT1和RT0，进入下一帧；**

**第3帧：**

**Drawcall 1->将RT0作为输入，渲染到RT1，交换，进入下一帧；如此往复…**

**虚幻4流体模拟教程(二) – 波动算法**

**Webgl-water Github**

****