**事件延迟**

Retriggerable Delay和Delay的区别

第一个你再次调用时他会重新记录时间，而Delay不会

**字符串的操作**

Append 添加操作

**还有很多类似的函数以此类推**

Buildstring（boolean）

Parameter 一个字符串

一个字符串

bool 打印true 或者false

String

**流程控制（循环语法）**

Branch 相当于if else

Do Once 执行以此以后就不再执行，除非Reset重置一下 Start Closed 开始时就关闭（一般用于执行一次，比如播放音乐）

Do N 执行N次

DoOnce Multiinput 这个只是多了几个节点

FIIp Flop交换执行A和B（用于灯的开关）

Forloop和forloopwithbreak =CPP中for

Gate four parameter is 交换

MultiGate 按顺序输出或者随机睡出

Sequence（顺序） 同时执行三个事件 并发（可以优化连线，可以防止一个阻断，导致的后面不执行 好东西）

**数组 相关操作我们可以在工具中找到**

Get 可以获取某个值

Set 设置数组元素

Add 添加元素

Add Unqiue 不能添加相同元素，就会添加失败

Inset 插入元素

Remove index 根据索引删除元素

Remove item 根据相同的值都会删除

Find item 查找元素 返回元素位置、

Contains item （包含数组）如果存在就返回true

Length 数组长度

Last index 最后元素位置

Resize 重新定义数组长度

Clear 清楚

Shuffle数组打乱

**结构体**

Break 把我们成员暴露出来，可以展开结构体

蓝图进阶

蓝图继承 和实例化

蓝图编辑界面中Construction Scrip（脚本） 比Event Graph早运行

Construction Scrip（脚本） 他可以实时的改变场景中的变化，因为在我们编译的时候，就会调用他，所以我们遇到改变场景中的TEXT时没有发生变化，就可以把它移到Construction Scrip（脚本） 中。 实时更新。

蓝图事件可以在函数内部调用，但是不可以在内部创建事件，因为函数在未运行时，作用于不存在，所以我们可以调用事件，而不是创建。

事件的调用我们可以直接执行该事件，或者利用命令行 Execute Console Command

关于这个命令行 param填写是ce+ +事件名称。就可以执行了

底层解析这个命令 发现是一个自定义事件，所以就在全图找这个自定义事件

关卡蓝图的交互（通俗一点就是让我们的actor动起来，有交互）

当我们选中某个actor时候我们就可以 在关卡蓝图中反键找到他的引用（小技巧）

Actor的移动我们可以使用addactorLocalOffset 其他的actor操作以此类推

关于actor接收问题，如果你没有将Input 下的receive input 设置成player0，他就没办法去接收玩家输入，所以这点比较重要

所有actor蓝图想要引用场景中的物体，就需要将该actor实例化

获取actor的名称 get display Text

Actor实现通讯

我们有时候没法保证一个类实例化了，所以我们可以利用character一定有这个实例化，所以我们在character中创建两个变量，然后我们在相关actor中 get player pawn 然后我们在转换为thirdpersoncharacter 获取两个变量。

获取场景中该类的所有变量

GetallActorOfClass（这个是获取该类的所有父类，如果我们创建了该类的三个实例化，所以他返回包含三个他的数组）

蓝图类和关卡蓝图之间的通信

Timeline 节点 再某一段时间内不断的去执行某一节点

我们可以在蓝图类中使用命令行调用

在我们创建变量中有一个叫做Event Dispatchers(调度机)这个是，也就是我们cpp中的绑定。

Call去调用绑定事件

Bind绑定一个事件，然后我们就可以利用他在关卡蓝图中调用（推荐这种）、

Unbind去解绑定

利用绑定实现蓝图类和蓝图类之间的通讯

摄像机

我们使用set view target with blend 切换摄像机