**集合特点：**

一种数据容器，一种数据结构

容纳多个数据，大小可变，空间不一定连续

**集合两大体系：**非泛型集合，泛型集合

**命名空间：**

非泛型集合 System.Collections

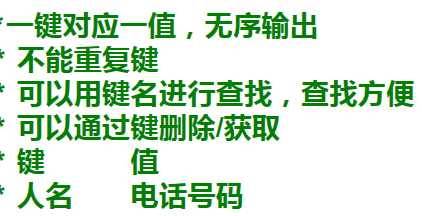
泛型集合 System.Collections.Generic

**非泛型缺点：**

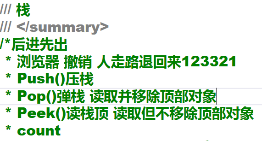
1. 性能不好，因为可能发生装箱。
2. 类型不安全，可能会发生类型转换的异常。
3. 使用不方便，用的时候需要手动做类型转换。

列表 ArrayList List< T >

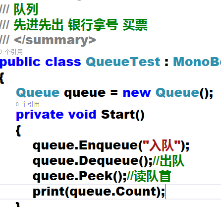
字典 Hashtable Dictionary< TKey,TValue >



栈 Stack (后进先出) Stack<T>



队列 Queue(先进先出) Queue <T>



# 角色换装案例

**需求分析：**

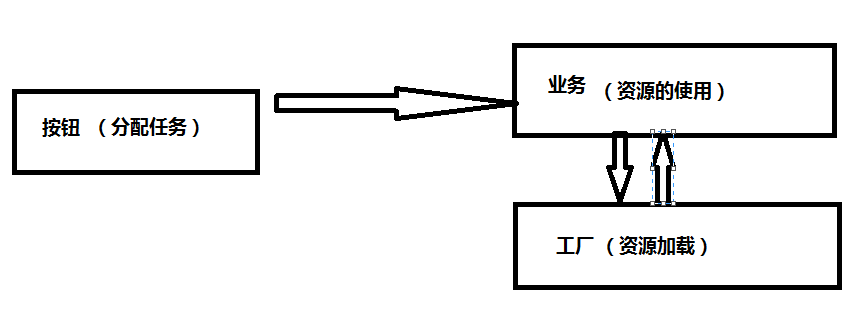
* **角色选择**
* **服装选择**
* **武器选择**
* **动画选择**

**技术分析：**

* **点击按钮实现切换**
* **按钮功能分为四类**
* **不同功能对应不同美术资源**

**问题：**

* **如何管理按钮点击（按钮只负责输入，不负责具体业务处理）**
* **如何组织功能（业务单独处理）**
* **如何管理美术资源在场景中的使用（资源存放，动态加载）**



# 常见寻路方法:

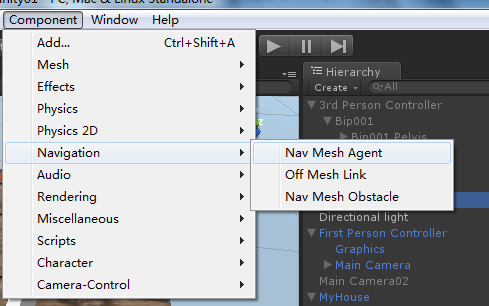
* 路点寻路
* 单元格寻路
* 网格寻路

# 实现寻路步骤：

1. **将场景中不动的物体勾选static，**

**到window中调出视窗，点击Bake，形成寻路（蓝色）网格。**

1. **需要自动寻路的物体，添加自动寻路组件。**

****

1. **添加脚本**

****

# NavMeshAgent属性

**Radius** **寻路的碰撞半径**

**Height寻路的碰撞高度**

**BaseOffset 寻路碰撞的位置**

**Speed 寻路物体的速度**

**Acceleration 转弯时的加速度**

**AngularSpeed 转弯时物体的角速度**

**StoppingDistance 停止的距离**

**AvoidancePriority 躲避系数**

# 寻路路径烘焙属性

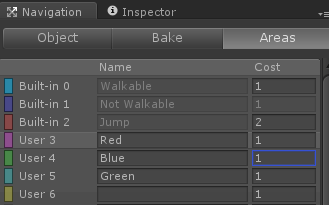
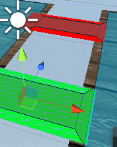
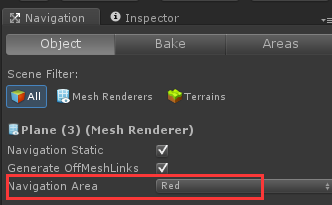
**Radius 是指寻路区域与障碍物之间半径**

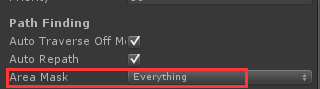
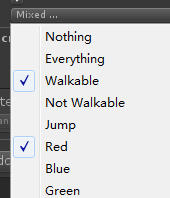
**Height 是指寻路区域与地面之间的高度**

**MaxSlope 是指寻路区域烘焙的最大坡度**

**StepHeight 是指台阶高度**

# 寻路系统区域遮罩：

1.  **分别添加自定义区域，如Red、Blue区域**.
2. **选择场景中的静态路面 指定到相应寻路区域****。**
3. **Bake寻路路面。**
4. **找到需要寻路的物体，设置可在寻路路面行走的区域。**

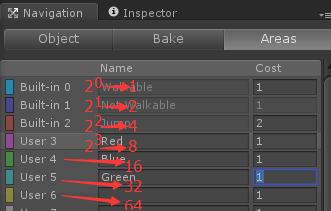
（PS：Cost：寻路区域消耗度，数值越大，从此寻路区域消耗越大。

寻路物体在区域消耗数值一样的情况下，会选择最优（最近）路面寻路，但如果寻路区域的消耗数值不同，会根据消耗的数值，越小越最优进行寻路。）

# 通过代码实现勾选不同的寻路区域：

**GetComponent<NavMeshAgent>().areaMask =9;**

**寻路区域每一区域都是2的幂**



**9则为Walkable区域（1）+red区域（8） = 9**

**Everything所有区域-1**

**Nothing任何区域都不能寻路 0**

PC端播放视频

1. 安装QuickTime播放器。
2. 不支持FLV格式视频。

# Mecanim新特性

* **针对人形角色提供了一种特殊的工作流，包含Avatar的创建和对肌肉的调节。**
* **动画重定向的能力，可以非常方便地把动画从一个角色模型应用到其他角色模型上。**
* **提供了可视化Animator视窗，可以直观地通过动画参数和Transition(动画过渡线)**

**等管理各个动画间的过度。**

# Mecanim动画系统常见术语

**Animator** （Mecanim动画系统组件）

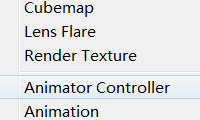
**AnimatorController** 动画控制器（动画控制器资源相当于一个容器，我们可以通过创建多个动画节点，并形成动画状态机，来处理各个动画之间的过度以及播放逻辑）

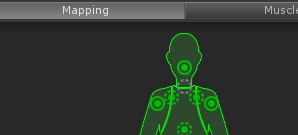
**Avatar**(将角色的骨骼转换为Mecanim可识别的骨骼)

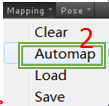
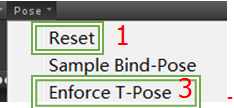
**Retargeting**（角色之间动画的互用也可以说是动画重定向）

1. **选择到模型源文件查看**
2. **将动画模式调整为Mecanim可识别样式**(人类，动画可以复用) （非人类，动画不能复用）

**Animation动画样式为**

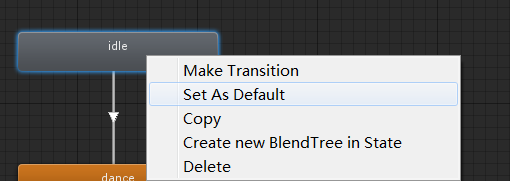
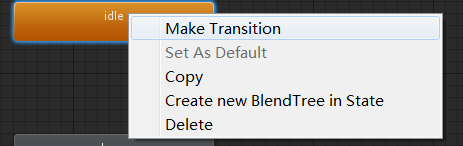
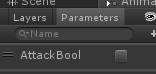
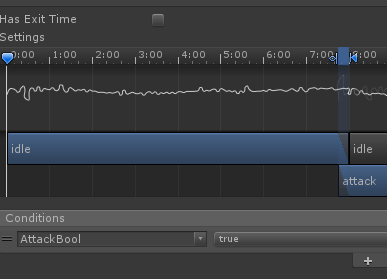
1. **Apply应用到模型**
2. **（若为对勾显示则骨骼匹配成功，若为叉号显示则需要重新匹配）**
3. **创建->给到角色上**
4. **找到Window下Animator面板 将需要的动画拖到面板中即可**

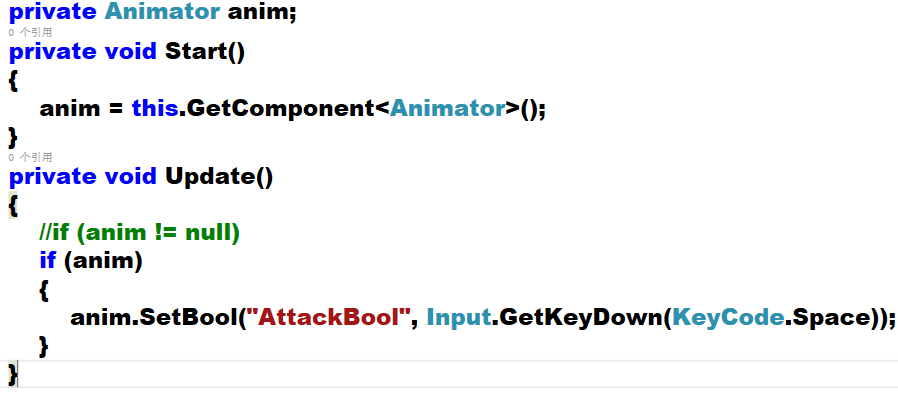
**需要重新匹配点击->进入此界面**

**找到->**

* **动画状态机动画切换**

**此颜色的动画状态，说明此动画为默认动画（此处跳舞为默认动画）**

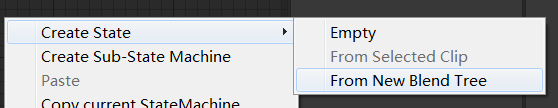
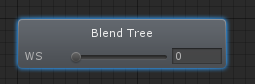
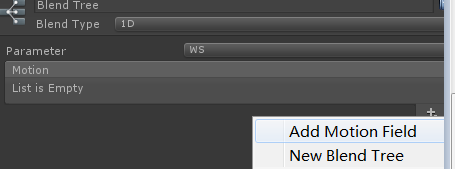
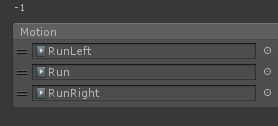
1. **若想实现角色从待机切换到跳舞效果，先将待机动画拖拽到Animator面板（或者右键创建Empty，将右边属性motion选择待机动画）-> 将待机动画设置为默认动画（在待机动画状态右键选择****） ->** **待机动画即为默认动画 -> 在待机动画状态右键选择****->**
2. **添加动画参数**
3. **选择动画过渡线设置条件**



# Mecanim BlendTree(混合树)

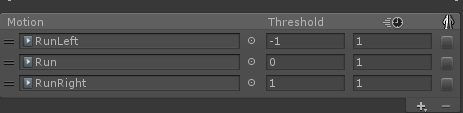
# 待机->跑0、左-1右1跑（水平轴）

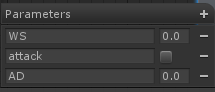
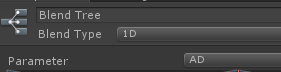
**（可实现一个动画状态有多个动画效果，如跑，既可以前跑也可以左右跑）**

1. **创建 BlendTree**
2. **双击blendTree动画状态**
3. **进入此界面，然后点击添加需要的动画->**

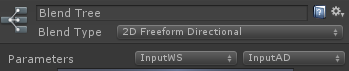
**注意：如果勾选则Threshold数值不能自己调整。**

**如果需要调整则取消勾选即可。**

**设置为，此处想通过键盘平行轴返回值控制左右跑动画，则为-1 ，0， 1。**

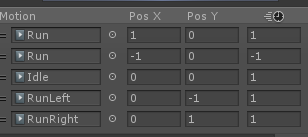
**添加AD小数，在BlendTree应用AD**

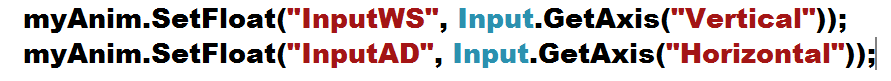
# Mecanim BlendTree 2D



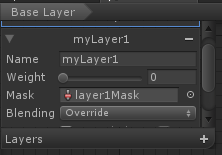
通过2个动画参数控制动画切换

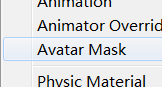
InputWS InputAD分别对应POSX 与POSY两个数值

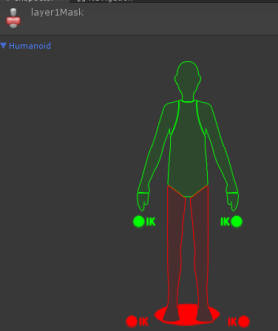




# Mecanim层设置，用于角色动画混合效果（如：边跑边攻击）

**点击此处添加层，可更改Name，通过Weight设置动画权重，Mask控制动画混合位置，并将此层需要的动画拖到面板中。**

**注意Mask 在Project视窗创建，选择**

**此处为将腿遮罩，则腿不会播放dance动画，当调节Weight动画权重数值时可看到融合效果（范围从0~1），1为完全融合。**

* **Weight 权重**

**权重是指某个动画层在整个动画中的影响，**

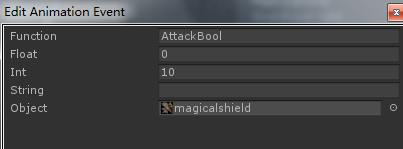
**若权重值为1则此层动画将与整个动画融合**

**若权重值为0则此层动画与整个动画完全不融合**

# 动画事件

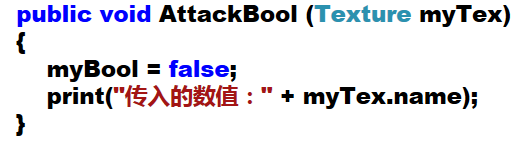
**找到角色源文件->找到需要的动画->**

调用的方法名

****

可以传的参数类型

**此脚本给在角色上，当攻击动画将要播放完时调用AttackBool（）；**

****

# 判断动画是否在播放

**Animator anim ;**

**//默认层的待机动画 哈希值**

**int idle = Animator.StringToHash("Base layer.Idle");**

**AnimatorStateInfo currentBaseStage;//当前播放的动画**

**void Start () {**

**anim = GetComponent<Animator>();**

**}**

**void Update () {**

**currentBaseStage = anim.GetCurrentAnimatorStateInfo(0);//当前播放的动画**

**if (currentBaseStage.nameHash == idle && !anim.IsInTransition(0))//正在播放待机动画，并且没有动画衔接**

**{**

**anim.SetBool("hit", false);**

**}**

# 布料

**Unity4中, 需要对SkinnedMeshRenderer使用SkinnedCloth, 或者对Cloth Renderer使用Interactive Cloth, 很麻烦, 在Unity5中, Cloth合二为一了.  
Cloth只能必须和Skinned Mesh Renderer搭配使用, 但是这不代表使用简单的物体时还必须在Max中导出一个带有蒙皮信息的FBX, 其实可以新建一个GameObject然后赋予Cloth组件, 这会自动添加Skinned Mesh Renderer组件, 然后在Skinned Mesh Renderer组件中赋予基本体的Mesh上去并且设置正确的材质也完全可以.  
布料可以接受外部影响,但完全不会将影响赋予外部刚体, 换句话说, 布料系统的物理模拟是单向的.58.png (1.8 KB, 下载次数: 1)**

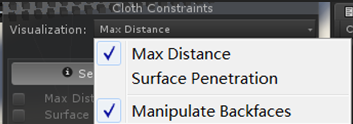
**[下载附件](http://www.element3ds.com/forum.php?mod=attachment&aid=MTk0NTEzfDU2N2FmMmM1fDE0NTk5MTIwMTZ8MHwzNjk0MQ%3D%3D&nothumb=yes" \t "_blank)**

**2015-6-18 17:30 上传**

# ClothConstraint布料约束



**MaxDistance可以设置每个顶点最大可移动距离,最常用的用法是将不能动的顶点的Max Distance设置为0.  
Surface Penetration表面嵌入控制，控制的是顶点最大可以嵌入到Mesh里面的程度. 在布料网格顶点比较稀疏的情况下可以明显对比出差别.**



**Visualization这里能够选择当前在Scene视图中预览Max Distance还是Surface Penetration. (这两者二选一), 还能选择是否让操作影响视口背面的顶点ManipulateBackfaces.  
Select编辑模式要先通过框选, 或者Shift+点击来多选, 来选中顶点, 然后勾选Max Distance或者Surface Penetration前面的复选框, 代表我现在要改变选中的顶点的值了。然后再在后面的数值里面输入想要的数值就可以了, 要想将当前有数值的顶点设置为Unconstrained, 只需要选中那些带有数值的顶点然后将对应复选框取消勾选即可。**

# 布料属性介绍

**Stretching Stiffness---拉扯硬度.  
  
Bending Stiffness---弯曲硬度.  
  
Use Tethers---默认开启, 绳索属性，两个固定点之间的属性  
  
Use Gravity---是否使用世界重力.  
  
Damping---阻尼会应用于每个布料顶点. 要想打造看上去抖动更小的布料, 可以调整这个属性.  
  
External Acceleration---恒定加速度（模拟风效）  
  
Random Acceleration---随机加速度.（模拟风效）  
  
World Velocity Scale与World Acceleration Scale共同组成布料运动时的物理模拟造成的影响比例.**

**Friction---当布料碰到在这个列表中存在的Collider时所产生的摩擦力, 这只会影响布料的模拟. 上面说过了布料的物理模拟是单向的.  
  
Collision Mass Scale--- 碰撞质量   
  
Use Continuous Collision---使用Continuous Collision, 增加消耗, 减少直接穿透碰撞的几率.  
  
Use Virtual Particles---添加一个虚拟粒子碰撞每个三角形来提高稳定性**。 **Solver Frequency---显然是一个优化参数, 默认120很高了, 可以试着调低一些.  
  
Sleep Threshold---静止阈值.**

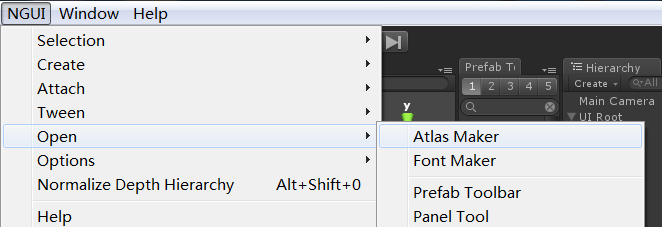
**出于性能的考虑, 可以对Cloth产生影响的Collider只有两种, SphereCollider以及CapsuleCollider, 但是SphereCollider可以赋予两个为一组(当然了还可以只赋予一个, 第二个保持null), 那么通过组合这两个SphereCollider可以出现第三种, Unity会将这组的两个SphereCollider进行相连, 那么就可以有圆锥形的Collider**

图集 精灵 锚点

# 图集制作Atlas

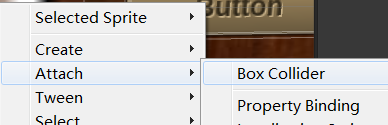
**精灵图集->**

**制作图集：**

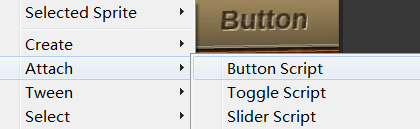
1. ****
2. ****
3. **选择需要的图片**
4. **点击Create即可形成图集**

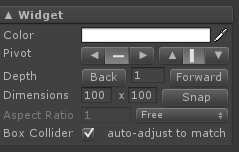
# 按钮制作：

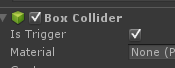
1. **创建Sprite，****选择Sprite，鼠标右键选择Attach，添加BoxCollider**



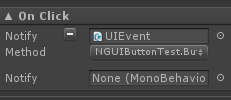
1. **选择Sprite，****鼠标右键选择Attach，添加ButtoneScript**



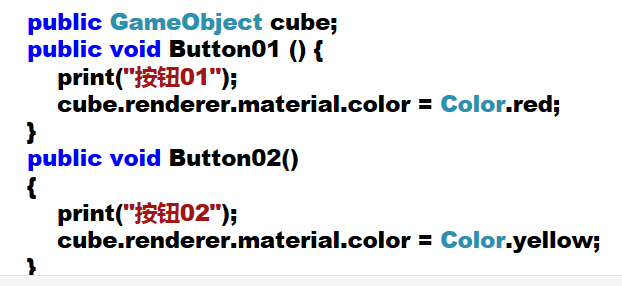
**注意：如果是通过在unity中AddComponent，将BoxCollider勾选**



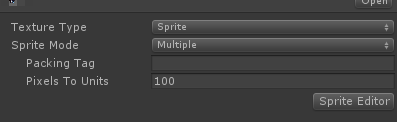
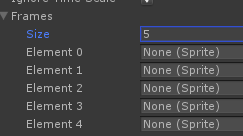
* **实现按钮点击事件：**

**找到按钮->把带有脚本的物体放到**

**->在Method找到要调用的方法。**

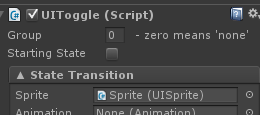
****

# 2D精灵动画

1. **创建2D精灵**
2. **添加 UI2D Sprite Animation组件**
3. **将图片设置为精灵模式，并且为可切割的**
4. **到->,将图片切割好，Apply。**
5. **最后到将精灵依次给到相应位置即可。可通过调节速度。**

# 开关制作

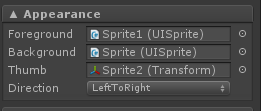
1. **创建精灵->Attach BoxCollider->** 
2. **在子集中创建精灵（选择点击开关时显示的图片）**
3. **将子集精灵拖到父级相应位置，**



1. **StartState是指开关的初始状态。**

# 滑竿制作

1. **创建****Sprite，给深度值较高的精灵，添加BoxCollider，添加****。**
2. **创建Sprite作为子集**



**Foreground前景图（此处当鼠标拖拽滑竿时，会有伸缩效果）**

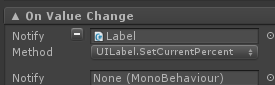
**Background背景图(固定不动)**

**Thum(此处创建一个精灵拖到此处，可形成游标效果显示**

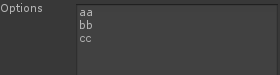
**)**

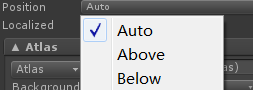
1. **Direction滑竿方向**

# 滑竿数值百分比显示

1. **找到滑动条组件，****，将创建的UILable拖拽到**
2. **调用UILable.SetCurrentPercent方法**

# 下拉菜单制作

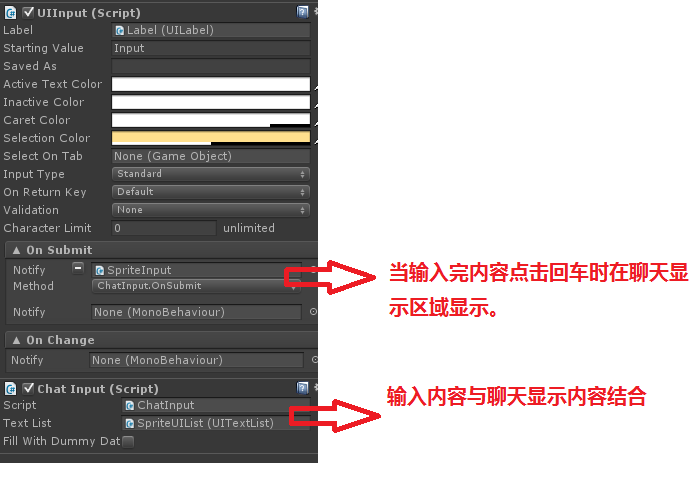
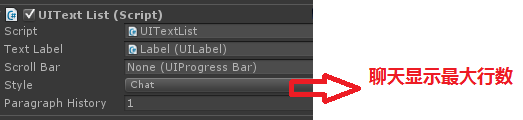
1. **在背景精灵中添加**BoxCollider ,以及**组件**
2. **将需要的下拉菜单选项输入Options中**
3. **将Lable拖拽到，（作用：弹出菜单选择的内容将显示在Lable中）。**
4. **默认按钮Labe内容。**

**设置菜单按钮自动，向上，向下弹出。**

# UIPlayerTween组件使用：

1. **创建一个精灵，给精灵添加碰撞，并添加PlayTweenScript组件。**
2. **录制一组Tween动画，录制好后，将Tween动画组件隐藏（之后会通过点击精灵控制这组动画的播放）。**
3. **将需要控制Tween动画的UI放到PlayTweenScript组件的位置**
4. **PlayTweenScript组件中为触发动画的方式，设置成为点击触发。**
5. **动画播放的方向：来回方向播放，正常方向播放，相反方向播放。**

# 聊天

1. **创建输入区域以及显示区域**
2. **给输入区域添加ChatInput组件**
3. **给聊天显示区域添加UITextList组件**
4. **若需要滑动条，可放到ScrollBar位置。**

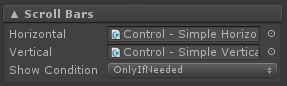
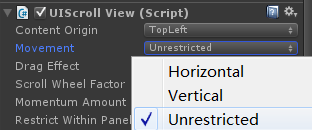
# 定制滑动区域

1. **创建背景精灵。**
2. **创建****（可滑动区域）作为背景精灵子集（利于管理）。**
3. **创建物品精灵，作为滑动区域的子集。**
4. **给物品精灵添加BoxCollider以及可在滑动区域滑动组件**

****

**若也需要拖拽滑动区域背景时有滑动效果，也可给背景添加碰撞以及滑动区域拖拽滑动组件**

# 滑竿控制滑动效果

1. **将滑竿拖到场景中**
2. **将垂直/水平滑竿放到****组件中相应的属性中****（若只需要水平/垂直，只拖拽一种即可）**
3. **若既想实现水平，也想垂直拖拽，不要忘记设置****任意方向滑动。Custom 是自定义滑动方向及偏移量。**

# 精灵拖拽效果

1. **给物品精灵添加BoxCollider,添加拖拽组件**
2. **若需要拖拽的精灵在所有UI最前方显示，创建深度值较高的panel并添加**

****

# 拖拽的物品栏排序

1. **创建Grid（并注意形成父子关系）。**
2. **设置Grid为平行布置还是垂直布置**
3. **设置物品栏之间间隔。**
4. **如何排序。**
5. **对齐方式。**

**是否有平滑动画。**

# 留在某个精灵背景上

**将背景精灵添加拖拽容器组件。**

# 将停留的物品进行排序

1. **创建Grid,并拖到容器组件的****位置。**
2. **设置Grid位置以及间隔距离。**