TechnologyArt-License

● 安装

● 将插件文件放置到引擎根目录下的 Plugins/Marketplace 文件夹中(如下图所示)

软件	(D:) > UE_4.27 > Engine	> Plugins > Marketplace			~
^	名称 ^	修改日期	类型	大小	
	CharlesFrameWork	2022/7/9 16:06	文件夹		
	ObjectDeliverer	2022/3/9 21:06	文件夹	18	

• 在引擎界面右上角"设置"中找到"插件"项



● 搜索找到 TechnologyArt 插件,并勾选 Enabled (如下图,需要重启项目生效)





● 在引擎界面上方菜单"设置"中找到"项目设置"项

	🔅 设置 🗸
	场暑设署
	0°2
◎ 🖡 项目标签 🔺 🖉 插件	
▼上 Scene(编辑器)	修改当前加载项目的设置

● 在项目设置页面左侧找到"Dev Kits Config"栏,并在其页面中找到"交互图表"项

U RAT MAR NO IN AL	a ∯-x0:#uxaz ×		- 0 X
сяна			
商日	- 按理 - Day Kite Coofig		
-70 E	- AFAC - Dev Kits Connig		INCOME.
110			40
392	· ARCREARCONNECTIONS Sting will, CERTICAL,		
205			
用数硬度 数尺			
苏持于总		ा आज सर 💿 🖞	
Succession Tass			
游戏			
AU 74		0 REALES 💿 🔂	
220030			
Dry Kits Conto			
引擎			
231.00	Detvis	-	
93342313	+ + CONTRO Met Work		
元秋万法	6.CH		
342	DRP0		
ARRENESER	D.KHQ	man /2000	
11200	0.000		
* C # 1	WARADP		
598	ALCON .		
	* 101NE MAR		
		i i i i i i i i i i i i i i i i i i i	
	w + /15/00 DV		
	Fait/485410000195		
	- CONVELUE.ope		
68		z v	
抗果性液液		were CE BES	
SLA BEERDOWA	> UV080		
100000000000000000000000000000000000000			
0.000			
(正府連載(赤池))			
用品語用			
N/5-82			
LOWINHA			
5405890389168			
World Partition			
编辑器			

点击交互图标右侧下拉箭头,在弹出菜单中点击"UILogicTool"新建空白 UI 管理
 器

	410	
化和浓度	(Q ===)¢.
项目	- 游戏 - Dev Kits Config	
		9ti 9A
	▲ 3直投業並存在ElefantSerkesSetting nill, 它自有可可入。	
用助纸法 影片		<u>•••</u> @ b
		osaites O ù
3270		
8278		0 mm iAm ⊗ 🗘
07530		
Ory Kits Cavilia		population of the second s
31/28		
51章		Abit Utopefeed
12.03.02		F: 2H URogic Teel
348763	··· CONFID Not Werk	Tig Kut
233.5.8	0XVA	Deviso = 200
18.0	BRDA	120/22
AGARRENTER	DKSI	
2028 15 Hotel 17	0,004	Projectili ogiciliai
2+0.0	UKBERD .	Utage tool
E83	00000	
812		
223852		
10.000		
7.0		
0.10		18
供熟菇准藏		
20.0. REFERENCE		
Rt		
12/28/11		
送於果我(未約)		
Cheel 新用目		
ALTYPIC RIA		
24%武器		
WWWWWWWW		
编辑器		

● 在弹出的资产保存窗口中,自定义一个保存路径及名称

u			资产另存为				×
Q 搜索文件夹	Q 搜索资产						in (†
▶ 💼 UMG ▶ 🛅 引擎		Blueprints	Developers	Maps	UMG		
	510						
路径: /All/Game 命名: <mark>NewUILogicTool</mark>						保存	取消
● 完成新建 UI 管:	理器及配	翌					

● 配	置序列模板	
🗊 🖈 and an in	t #lio ♦ #1012mm ×	- 0 >
659 3		
项目 11版 15.000 15.000	্য আজ • ঠিয়ট - Dev Kits Config DeviceConfig এ এইএইছার্জেরিএস্টেএএইডেম্বেডালের্চানট, চেউচাল্মস,	- Mari INA
NA RNRM RN NM	w ∑iii File loos Tempioni fort 56; Post- Tempiolis MAG	
Senselection 結況	UMDRI de Treydele Paserent Casses	n Differance
地合新規具 Dev.Krs.Corfig 引擎	v Batiliki Pranot M Age Wendon Uwiteraan	инон Т 200 0 Салина (Салина) (Салина) 10 Салина (Салина) 10 Са
(1991:00) 日秋日秋秋 日秋天秋 1991	Debug • - 00 MIC her Nook BJS200 BJS200	
2000 送出接角线控制度 助用 动用使的性	D (अंग D (अंग D (55) 20	
<u>大元(17)</u> 500冊 天時 既任) 전체주석 (145年2년 · • - CONNEC MAR · · · · CONNEC MAR	
120192 12014 12:2102		
3626	· - CONTROL BLogie	
成長前提数 成長前提数 成人		
RHEACONN Hà	► UTA26 ► ATHERE	

● 为项目配置 GameMode(必要)

● 在项目设置页面左侧找到"地图和模式"栏,点击后在右侧找到默认游戏模式项

1 217 898 810 IR	सः: 			

商日	- 15日 #h回印##+#			
「「日	*坝目 - 地图构模式			Internet Internet
110 1000000	RECORD, BOUGHAUNTEDDES, MARKEN			40.0
ALE .	· CAUMERCONCEPTION CONTINUES			
78.0	× 8×85			
D.1085.0	B-LDB/RZ		Lanckielder V 🖉 🖬 🛈	
8H:	> 2072 01182.			
\$17.0	w RURT			
GeneplexCesa			terre v	
游戏	加速等于比较常		<u>▲</u> e ∎	
it.t.a	ORESISTER .		omuts ⊙ 🕁	
透开展课题			Serve V	
Dave Hitte Section	8/08/13/00		🚾 e 🖻	
引擎	> 80			
<u>69100</u>	₩ \$182 Å			
习机程程统	\$175 F			
用加系统	的名词复杂的名词			
3.00	R-BURYON P		141(12) V	
3933929302788	DEMANDER		40 V	
(1.000	* 2028			
关系的历	223455		inveksione v 😤 🖪 🕢	
9.99.M				
88				
89				
12338.0				
10.0000				
118				
RM				
然熟谙准款				
HA.				
REENERGY AND THE REENERGY				
2538				
812				
演奏				
建装置数(水物)。				
用類				
内に設備				
Chaos #F#F#				
Gameploy/RICE				
SmiRN				
Wards Partition				
编组器				
▲ 上	上 て 社 迩 乳 光 抑 書			上十沉里斗醉门游声带
■ 黒ゴ	日下世面头井椤袋	Dev 化到 BF		二一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一
2				



● 为项目配置输入映射(必要)

● 在项目设置页面左侧找到"输入",点击该项,并在该项页面右上角找到导入按钮。

u _ xeet			
所有设置	我您详情		•• ۵
项目	▲引擎 - 输入		
	输入设置,包括默认输入操作利坐标给绑定。		- 号出
地图和模式	♀ 这些设置被保存在Defaultinput Ini中,它当然可写入。		
2018	4 帰定		
用成硬件	操作和编映射提供了一种机制,通过在输入行为和激活该行为的按键之间	重人一个间接层,以便利将模和编映射影像人行为,操作块制针对该下和松开按键,而编块制制针对拥有连续范围的输入。	
影片	18471938 🕂 👩		
	141495 + 🖻		
GameplayTags	调整映射		
游戏	4知口居住		
	ADDING		
資产管理器	難以祝口服杯捕获模式	飘和我下的进行拥获 ▼	
Dev Kils Colling	股认现口服标锁定模式		
引擎	4 除动平台		
层级LOD			
<u>等款网络钟</u> 异航系统			
地形		•	
调试摄像机控制器		Defautivitual.loysticks -	
均置	ALCORD THE	a +	
大下序列 体统器			
教程	▲ 南與键盘(移动平台)		
拉制台	使用自动修正	•	
垃圾回收			
10.05		Playeritiput 💌 🔶 🔎	
群集管理器		InputComponent 💌 🔶 D	
▶驗人	▲ 投制台		
数据驱动CVars File		1数编元家 🕇 會	
2000			
淮南			
100 JA 201 BB / 10 HB \			

● 点击导入按钮,在弹出窗口中,找到插件安装路径,并在 CharlesFrameWork-Input 文件夹下找到"TA_INPUT.ini"文件,双击该文件完成导入。



TechnologyArt-插件预设类

● 基本概述

插件提供预先设计好的功能类,帮助用户快速实现常用基础功能,可在多个项目中快速复用。预设类与 API 高度联动,通过 API 可完全控制预设类。当用户有自定义需求时,仍可继承或覆写预设类。

注意:如需对预设类进行功能拓展,建议将插件中预设类的原始文件复制,并将复制 文件移动到个人文件夹中,再对复制的类进行功能拓展。

● 预设角色类

● 在左侧"放置 Actor"面板中可以快速找到插件预设的 actor 类,拖拽到场景中即可 使用

向 内容浏	览器 📫 放置Actor 🛛 🗙
Q 提索	
	IEACHNULUUT ANI
	BP_DevKitCharacter
÷	BP_DevKitFlyPawn
•	BP_DevKitPawn
	BP_PopUpMark
	BP_QuadTreeCounter
	BP_QuadTreeMark
	Kit Mark Actor

 如需对预设类进行自定义修改,可在内容浏览器右下角视图选项中勾选"显示插件 内容"。



 在左侧文件路径中找到"GameFrame"文件夹,文件夹中可找到三个预设角色类, 可复制到个人文件夹中进行自定义修改。







输入按键	执行操作
鼠标左键	控制相机围绕中心点的角度操作
鼠标右键	控制相机中心点平移
鼠标滚轮上滚/下滚	控制弹簧臂伸缩
键盘 WASD	控制相机中心点平移
移动端触摸点击	控制相机围绕中心点的角度操作
移动端两指缩放操作	控制弹簧臂伸缩
鼠标中键	控制中心点垂直移动

● 以上按键所对应的执行操作可通过 pawn 的参数面板进行配置

🔀 细节 🛛 🗙 🌍 世界场景设置			
🛓 BP_DevKitPawn		十添加 • 🖬	~ n
🛓 BP_DevKitPawn (自我)			
✓ ▲ Root (Root)		在C++中	编辑
🔫 🛓 Origin (Origin)		在C++中	编辑
🔻 💣 Spring Arm (SpringArm)		在C++中	编辑
■< Camera (Camera)		在C++中	编辑
(Q 搜索			* 🕸
通用 →TA Actor 杂项	流送所有	ī	
▼ * TA Mouse			1
左键功能	旋转视角	×	
右键功能	移动相机中心	~	
滚轮功能	伸缩臂缩放	~	
左键双击功能	关闭	~	
▼ * TA Config			
Rotator Switch			
Translation Switch			
Scaling Switch			
▼ * TA Rotator			
Origin ZRotator Range	-360.0	360.0	
Origin YRotator Range	-89.0	89.0	
LMB Target Rotator	0.0 0.0	0.0	
Rotator Speed	18.0		÷
Rotator Proportion	15.0		6
▼ * TA Movement			
▶ XRange	-10000.0	10000.0	\$
YRange	-10000.0	10000.0	\$
▶ ZRange	-10000.0	10000.0	6
Output Zoom Range	50.0	100.0	\$
Location Speed	2.0		¢
▼ * TA Init Property			
Target Zoom	4000.0		\$
🕨 D Initializa Botator	nn	1 00037 0 0	-

● 鸟瞰角色是一种围绕中心点旋转的相机,相机与中心点的距离由弹簧臂(Spring Arm)控制,中心点可进行位移交互。



● 将 BP_DevKitPawn(鸟瞰角色类)拖拽到场景中。



 在鸟瞰角色右侧细节面板中,可设置继承自 Camera 的基本参数,如视场 (FOV)、后期等。

① 细节	💿 🕒 世界场景设置
8 BP_DevKitPawn	1
→添加组件→	●8 编辑蓝图 -
搜索组件	Q
👌 BP_DevKitPawn (🗎	1身)
▲ ● Root (Root) (继承 ▲ ● Origin (Origin) (▲ ♪ Spring Arm (S 参 Camera (Ca	^氏) (继承) SpringArm) (继承) amera) (继承)
搜索详情	∙ ⊙ 🏢 🔍
⊿ 变换	
位置 🔫	-90.0 🔊 0.0 🔊 20.0 🔊 🖻
旋转 🗸	0.0 * 0.0 * 0.0 * 0.0 *
缩放 🔫	1.0 🔍 1.0 🔍 1.0 🔍 🖬
▲ Root	<u>e. s. au</u>
移动性	● 静态 ┆ 固定
⊿ 摄像机设置	
投射模式	透視 👻
视场	90.0 *
高宽比	1117IN (111) -
⊿摄像机选项	
约束高宽比 使用Pawn控制旋转	

● 下拉细节面板,在 MoveCurve 插槽中,可自定义或修改相机执行移动 API 时的 运动曲线。

	-		
Maria Curria	Ī	MoveCurve	-
Move Curve		φ	
A Down Property C	ontrollar Switch		

● 在 Pawn Property Controller Switch 中可选择是否锁定旋转、移动、缩放操作。



● 在 Pawn Property LMB 中可设置相机偏航角(ZRotator)、俯仰角(YRotator) 的限制范围,还可以设置旋转的速度。



● 在 Pawn Property RMB 中可设置鼠标控制角色平移的范围、平移速度、中心点 距离与平移速度的变化范围。

⊿ Pawn Property RMB				
▷ XRange	X -100.0	2	Y 100.0	2
▷ YRange	X -100.0	2	Y 100.0	2
▷ ZRange	X -100.0	2	Y 100.0	2

▷ Output Zoom Range	X 0.0	2	Y 500.0	2
Location Speed	5.0		2	
2				

● 在 Pawn Property Initialize 中可设置角色初始弹簧臂长度、相机初始角度、角色 Key 值(对应 API 中填写的 PawnID)、设置该角色是否为程序运行时默认使 用角色。

4	Pawn Property Initia	ılize
1	Target Zoom	2000.0
D	R Initialize Rotator	X 0.0 V 0.0 V Z 0.0 V
	S Pawn Keys	Main
	Main APawn	•

● 在 Pawn Property Zoom 中可设置弹簧臂长度在设置范围内的伸缩速度变化。

⊿ Pawn Property Zoom						
D Zoom Regional Range	X	300.0	2	Y	3700.0	2
▷ Zoom Speed Range	X	30.0	2	Y	370.0	2

● 漫游角色



输入按键	执行操作
鼠标左键	控制漫游角色前进与前进方向
鼠标右键	仅控制漫游角色视角
键盘 WASD	控制漫游角色前进后退左右平移
鼠标指向地面双击鼠标左键	角色自动寻路走向鼠标指向位置
移动端触摸点击	控制漫游角色前进与前进方向
移动端触摸双击	角色自动寻路走向鼠标指向位置

- 漫游角色是一种第一人称视角角色,可控制角色在水平面上的移动,拥有登上台 阶、重力等能力。
- 将 BP_DevKitCharacter (漫游角色类) 拖拽到场景中。



 在右侧细节菜单中,可设施视角旋转速度、角色 Key 值(对应 API 中的 PawnID)、设置该角色是否为程序运行时默认使用角色。

⊿ 属性值			
F Rotator Speed	1.0	2	
S Character Keys			
Main APawn			
Is Flying Mode			

 在角色移动:行走一栏中,可以设置角色最高能登上的台阶高度、最大能移动的 斜面角度、最大行走速度等。



● 飞行角色



输入按键	执行操作
鼠标左键	控制飞行角色视角
鼠标右键	仅控飞行角色前后左右平移,仅在鼠标指向物体时 可用
鼠标滚轮上滚/下滚	控制飞行角色与鼠标指向物体的距离缩短与拉远
键盘 WASD	控制飞行角色朝当前方向前后左右移动
键盘 QE	控制飞行角色垂直上下移动

- 飞行角色是一种第一人称视角角色,可控制角色自由飞行。
- 将 BP_DevKitFlyPawn(飞行角色类)拖拽到场景中。



● 右侧细节面板可填写 Pawn Key (对应 API 中的 PawnID)

⊿ 信息	
Pawn Key	

- 预设标签类
- 弹窗标签



- 弹窗标签预设类提供在场景中显示/隐藏标签及点击标签弹窗的能力,提供样式、 大小等 UI 接口。同时该预设类下可挂载任意自定义组件,如模型的显示隐藏、 粒子的激活与停止、自定义场景 UI 的显示隐藏等皆可以挂载在该类下,通过 API(ShowMarkAPI)对其控制。
- 在插件路径中找到"BP_PopUpMark"。

🧱 内容浏览器		
□ 添加/号入 → 四保存所有 ← → ► CharlesFra	meWork内容 ▶ Code ▶ Labels ▶	ĥ
寝 捜索路径	▼过滤器▼ 搜索 Labels	D E
■ ChaosSolverPlugin内容		
▷ 📾 ChaosSolverPlugin个C++类		
▲ CharlesFrameWork内容		
P ArtContent		
D ■ Code D ■ AnalysisTools	image Markwidgets Model Label Mark BP_PopupTip UMG	
CommonUI -		
▷ ■ DevDebugTool ▷ ■ Editor		
GameFrame		
▷ 🗀 Labels		
TouchCore		
🖻 🖿 Util		
LicenseForIM LicenseForPak		
Maps	7 项(1 项被选中)	④ 视图选项▼
● 将"BP PopUpMark"拖挡	 我到场景中。	



● 右侧细节面板中"TACONFIG"一栏中,可设置标签样式、标签渲染枢轴、标签大 小缩放、是否需要点击弹窗。

▲ * TACONFIG			
D Button Style			
▷ 枢轴	X 0.5	Y 0.5	2
Title	标题		
UIScale	0.2	2	
⊿ 默认			
是否需要弹窗	2		

● 在 Actor 栏中的标签插槽,可添加 Tag Name,该名称可用于 ShowMarkAPI 中的调用(对应 API 中的 Mark Name)。同一 Actor 可添加多个 Tag Name,可对应不同 API 调用同个标签显示隐藏。

⊿ Actor	
在该关卡中1个已选中	持久关卡
转换Actor	选择一种类型
可被伤害	V
初始生命周期	0.0
生成碰撞处理方法	固定生成,忽略碰撞▼
查看目标时寻找摄像机组	Z
忽略原点偏移	
可存在于群集中	
▷ 枢轴偏移	X 0.0 V 0.0 Z 0.0 V
Actor Guid	{0AB03C4A-4B4D-8B48-1085-C
标签	0 数组元素 🗕 🛨 💼
	A
⊿ 标签	1 数组元素 🕂 💼 🦻
ji 0	111 👻 🖻



● 密集撒点聚合标签



 密集撒点聚合,是智慧系应用场景中常用的标签功能之一,使大量标签聚集时可 聚合并显示堆叠数量。



● 在插件目录中找到 Actor 文件夹。



BP_QuadTreeCounter 代表标签样式与类型,可双击进入蓝图界面进行样式、参数设置。也可在左侧组件栏自定义所需挂载的组件。该预设类仅定义撒点标签样式,无须拖拽到场景中摆放。

11 DP: Stuad FreeSouncer 文件 編編 亦产 直看 明述 第ロ 利	0		🔭 🗕 🗗 🗙 🕮 : - Quad Tree Counter
- 2 相件		3. 信节	
+ 添加相作 - 調整 0	- 24 - 🔚 🔎 🐝 リュー・ 🔐 🜌 📐 - 末法中朝武功会-	波索译情	ρ 🔳 👁 -
BP_QuadTreeCounter(自身)	喻译 保存 测范 查找 除藏不相关 关设重 <mark>关系以值</mark> 运行 Wilaux通道		
4 /2 Mark Meth (markMeth) (18.92)	■ 我口 f Construction Scrip 業事件面表		
Count Widget (CountWidgetComp) (総宗)	☆ 🖕 📦 📑 BP_QuadTreeCounter > 事件图表 缩放-3	D KESA	X 0.5 Y 0.5
GAPIComp (APIComp) (總承)		UtScale	0.2 0
		Actor Tick	
			2
		Tick间隅(钞)	0.0
		允许开始播放的Tick	
			÷
A ROEM		****	
+ Ka 2 122 0 ···			
			Q + 100
4函数(24可加率)			-
↑ 构造脚本		∡ Quad Tree Mark	
▶接口			0 0
老 +			X 0.0 Y 0.0 Z 0.0 Z
4 空里 +			X 0.0 Y 0.0 Z 0.0
▷ + TACONFIG			
▶相件 Ten An DD Label Tin			0 0
- As or caller tup			• •
Marks			0 数端元素 + 會
事件分发器 +			х + Q + х
			0数41元素 + 商
			-
	萨奥		
	凶田		10000.0
	2. 编译图结果	∡ 夏8	
		仅与拥有者相关	-
		展定相关	

注意:添加新的组件后,必须将其拖拽到 Count Widget 下。

BP_QuadTreeMark 代表标签撒点位置,可手动拖拽到场景中摆放,也可自行拓 • 展动态生成逻辑。



- 撒点类型与聚合参数均可在"Quad Tree Mark API"中进行调整(详见 API 说明)
- 聚合标签开发拓展-如何获取未聚合时的每个独立标签对象
- 在 QuadTreeCounter 中,拥有以下参数 •

Quad Tree Mark				
Obj Count	0	2		
^v Center	X 0.0	Y 0.0	Z 0.0	2
Ext	X 0.0	S Y 0.0	Z 0.0	2
ls Leaf				
Max Depth	0	2		
Depth	0	2		
Child Nodes	0 数组元素	+ 🖻		
Quad Mark Type	None 👻 🦛	0+×		
Quad Tree Marks	0 数组元素	+ 🗇		
Childs Draw				
Has Child				
Draw				
Draw Alpha	10000.0	2		

- 其中 Obj Count 代表当前标签的聚合数量,即 Obj Count=1 时,则标签未聚合
- 此时获取 Quad Tree Marks 数组中的第 0 个元素即可获取未聚合的单个标签对 象



DataModel

- DataModel 是一个全局可调用的"数据仓库"。
- 在蓝图创建窗口中,选择"KitBaseDataModel"做为父类并创建蓝图。

U	选取父类	×
⊿ 常见类		
C Actor	Actor是一种可在世界中放置或动态生成的对象。	Ø
8 Pawn	Pawn是一种可以被"控制"的Actor,且可以接收来自 Controller的输入。	0
()角色	Character是Pawn的一种子类型,增加了可四处走动的功 能。	0
¥ 玩家控制器	PlayerController是Actor的一种子类型,其负责控制玩家所使 用的Pawn。	0
■ 游戏模式基础	GameModeBase定义了正在进行的游戏、其规则、得分以及 游戏类型的其他方面。	
Actor组件	ActorComponent是一种可复用组件,能被附加到任意Actor 上。	0
😪 场景组件	SceneComponent是一种组件,能够进行场景变换并可被附 加到其他场景组件下。	0
⊿ 所有类		
datamodel		×
⊿O _Object		
O HttpInfoDataModel		
KitBaseDataModel		
O BP_PopupDataMoo		4L + 10
4 坝(1 坝假选甲)	● 视图:	选坝▼
	选择 取	肖

● 创建后即可在全局蓝图中调用该 DataModel。



TechnologyArt-插件其他拓展功能

● 基本概述

插件拓展了众多快捷功能,包含美术、程序、UI 等多个方面,了解并使用 这些功能可提高制作效率。

● UI 控件功能拓展

● 控件事件

● 当控件父类为 Kit Base UMG 时, 控件蓝图将拥有显示隐藏接口

unimia / 🖂 TestoWo		
文件 编辑 资产 查看 讲试 窗口 桥的		父常: <u>Kit Base UMG</u>
X IBE		
		🕵 设计器 👌 🔫 🛃 😹
🚔 具的正面	■ 単件田表	
+ 版址 ▼ 祖宗 • ~	☆ ◆ ◆ InstUMG > 事件图表	缩放1:1
+ 秀田ト		and the second se
▲ 書 事件图表		
◆事件预构造	1. · · · · · · · · · · · · · · · · · · ·	
◆ 事件构造 ▲ 事件です。		
V IPT TICK		
A UI		
M On Show		
🚺 On Hide		
老 +		
92 +		
事件分发器 🕂		
Q HF		
推滚评情 🔘 🏭 👁		
∡ Kit Base UMG		
Ultype 自定义类型 -		
UM020rder 0		
4 95 3Q		
D. 副色和不适响器		
D 前景開急 🗾 総承		坎什萨凤
D 1872 0.0		JIII III III
▲设计器		
		ol #
REAL None	·昭凡组织动导作各称米超成51用	A Q

- On Show: 该事件将在该控件显示时执行
- On Hide: 该事件将在该控件隐藏时执行
- 右键点击 On Show/Hide 的实现事件将可覆写或拓展该控件显示隐藏时执行的逻辑。不实现事件时将默认继承父类中的显示隐藏方法。

🖞 Untitled 🔚 TeatUMG	× / / / / / / / / / / / / / / / / / / /	- • · ·
文件编辑 资产 查看 讲试 窗口 帮助		父問: <u>Kit Base UM</u> C
⊁ Ⅱ目栏		
🌺 - 🔚 🔎 📫 🎜		🔂 设计器 👌 📜 國表
A 我的孤国	🔐 專件的表 🛛 🖂	
+ 新潮 ■ 推察 🔎 👁 =	☆ ◆ ◆ TestUMG > 事件图表	
4因表 +		
▲書事件因表 ◆事件我构造 ◆ 事件相通 ◆ 事件相応	点击右	建新建节点
(函数(4)可要差)		
▲接口		
⊿UI		
On Show 大田和名 P2 On Hide 安規事件 家 なおりのます。		
空里 ····································		
事件分发替 🏭 复制 Chine -		
atoise 🙀		
将这个司	官员政府 均一个新的事件实现。	



 如需继承父类中的显示隐藏控制(由 API 控制),需右键事件点击"将调用添加 到父项函数",将该节点与事件连接。相反,如需覆写父类中的 Show/Hide 方 法,则无需调用父项函数。



- 勾选组(Check Group)
- 勾选组是一组勾选按钮状态互斥的 UI 控件,并使勾选后的按钮无法第二次点击 取消选中,仅可通过点击同组其他勾选按钮更换选中按钮,有效防止在当前状态 时重复调用当前状态事件。勾选组拥有选中、未选中的事件回调,极大的方便 UI 制作。
- 在 UMG 控件中, 搜索"Check"。



如图将控件按照层级摆放。

u Untitled	🔚 NewWidgetBlueprint	Luntitled	🖉 🔚 TestUMG+ 🛛 🛛 🛛	ALA ALA		
文件编辑 资产 查看 调试						
1913 - In NO		些中调试对象▼ 调试现验器				
😥 控制版					160 200 230 300 300	① 细节
check	×	,缩放+7				
▲ 週用 ○ 🗹 勾选框						
 Kit Check Group Kit Check Group Item 						
.≞ 居級						
· 搜索控件	Q					
▲田 [画布面板] ▲田 [画布面板]	ଳ ଏ ଜ ଜ	2				
▲ Ⅲ [水平框] ◆ KitCheckGroupItem_48	- 					
 KitCheckGroupItem_135 KitCheckGroupItem_189 	ല് അ ല് അ					
 KitCheckGroupItem_233 KitCheckGroupItem_267 	ല്യ ല്യ					
L						
📌 动画	📂 Bri (il) Suk	💫 编译器结果				
+前面 接索动面	[7867.84]Test [786					

注意: 使用 KitCheckGroup 必须在 Check Item 之上包裹一个面板控件,不限与水平 框面板, 画布面板、垂直框面板等均可。

● CheckGroup 仍有常规参数设置,其中勾选 Group Allow Switch 使勾选按钮可重 复点击以取消选中状态。

文件编辑 资产 產者 调试 窗口 帮助				: Kit Base UMG
💏 - 🔚 🔎 🔳	- <u>880-011/18</u> - 01/1288		🔀 设计器 >	- 图表
🗢 1280.66				
搜索投制板	📭 🥵 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉	Fel KitCheckGroup 75	Is Variable	打开KitCheckGr
 場当長 		現象は彼		
> 合成		1000 (NOVING 10)		~ ~
▽ 51300 ◎ 面板				
			18.4 ·	
≥特殊效果		位置× ····································	0.0	
▷ 週用 ▷ 明平		Retr	× 1000	
◇用户创建			\$ 30.0	
▷优化			× x 0.0 ¥ 0.0	2
● 杂项 h CaleDister			< 🔳	
P SCUI			s 0 🗳	
12 周段		∡ Kit Check Group		
搜索拉件	<u>م</u>			
4 [TestUMG]				
 Classification KitCheckGroup_75 		⊿ □) 切(の)住		
- mu (ok-+ 41)		重载可访问款认值		
 KitCheckGroupiter_135 			E3	
 KitCheckGroupItem_189 KitCheckGroupItem 233 			No.	
	₽ •	∡{fi为		
				- 837-
			< 🗹	御定▼
	未设置设备安全区		◆ 非可应中激试(仅自身)▼	御定▼
	1280 x 720(16:9) DPI缩放0.67 ↔	波泉不透明度	s 1.0 😴	
→ 和面	鮮 対荷軸 🛃 論律器結果			
→ 胡田 技術功能 ●	• [8524.36]TestUMG 编译成功:[52 蜀砂内](/Game/UMG/TestUMG)			

● Default Check index 可设置该勾选组显示时默认选中的 check 按钮序号, "-1"为 不进行默认勾选, "0"为默认选中勾选组中的第一个按钮, 以此类推。

ZOrder	st 0 🔹
∡ Kit Check Group	
Group Allow Switch	
Default Check Index	0 0

● 在控件蓝图中, CheckGroup 拥有以下封装函数:

Get Current Checked: 返回当前选中按钮

Get Current Checked Index: 返回当前选中按钮序号

Set Check Next: 设置选中状态为当前选中按钮的下一个,如已为最后一个,不做操作

Set Check Prev: 设置选中状态为当前选中按钮的上一个,如已为第一个,不做操作 Set Current Checked: 设置当前选中按钮

Set Current Checked Index: 设置当前选中按钮序号



注意: CheckGroup 中,按钮 Index 从 0 为起始 Index。

 在选中按钮 1-选中按钮 2 的操作流程中,勾选按钮按下及弹起事件顺序如下, 某些情况下需特别注意事件执行顺序。



注意: 当在 OnShow 事件中使用设置勾选状态时, 需在 set 节点前添加 Delay 节点 (延迟时间为 0.1 即可), 以错开生命周期。



- 控件蓝图通信
- 在全局蓝图中可使用 Get UIBy Class 进行控件蓝图间通信, 控件蓝图父类必须 为 Kit Base UMG。



● 编辑器功能拓展

MarkEditorMode

● 由插件拓展的快捷工具栏,通过切换"模式"打开 MarkEditorMode 工具栏。



注意: 在 MarkEditorMode 模式下时,编辑器交互被工具交互代替,这将无法使用选择模式下的功能,如选择物体等。注意不使用工具栏时将模式切换回选择模式

● 创建与使用视角文件



- 插件工具栏提供快速创建视角文件功能,调整镜头时,无须手动记录坐标、弹簧 臂长度、当前角色等参数,创建视角文件后,通过调用 API 指定视角文件,快速 完成镜头设置。
- 在使用 Dev Kit pawn(鸟瞰角色)、Dev Kit Fly Pawn(飞行角色)、Dev Kit character(漫游角色)的前提下,点击上方菜单"运行"按钮,运行程序。(如未 使用插件所带的三种角色之一,则无法在运行时记录视角文件)

文件 编辑 窗口	1 能助	<u> </u>																			
🐠 MarkEditorMo	ie 💠 😻 Billiaci				-	26	655		- 10				1155			1	👹 美卡		1 世界大纲视图		
测试按钮	新样条线	创建视角	APII编辑器	保存当前关卡	液円留理	根式	内容	#SIRM	10日 10日	TECHNOLOGY ART	Megascans	1101	试场动轰	NIR	通行	en ·	搜索			ρ÷	į
∡ • Action Setting				🔡 R(E) 1													林装	Untitled (1858)	8)	英型 。 11月15日	ł
设置编辑模式	关闭	*		▼ → 透視	: 1 % % 照	夏示					dittor !	* @	20	10	10. 0	.25 3 4		Atmospheric			



在 MarkEditorMode 工具栏下方"视角文件名称"一栏填写需要创建的视角文件名称,填写名称后点击"创建视角"。

🗤 MarkEditorMode 🏾 🎸 放置actor	
 MarkEditorMode 放置actor 选择对应的编辑模式进行使用,场景中; 	点击鼠标左键可快捷关闭 关闭▼ 关闭 打点工具 视角工具 祥条线绘制工具 AI行人工具 交通组件生成 车辆生成 API生成工具
✓ MarkEditorMode ✓ 放置actor	
选择对应的编辑模式进行使用,场景中点	ā击鼠标左键可快捷关闭
	视角工具▼
▲ * TA 视角	
New Camera View	
視角文件名称	
視角过渡时间	2
自定义Pawnid	Main

● 视角文件将被创建在内容下 CameraViews 文件夹内



运行时记录:记录当前 DevKitPawn 的坐标信息、伸缩臂信息、PawnID、过渡时间 未运行时编辑器视口下记录:记录当前视口坐标信息,伸缩臂长度将由当前视口角度 射线击中的碰撞为准。(当视口中无有效碰撞物体时,记录的坐标可能会在远处,建 议在朝向物体时再记录视角文件。

视角文件将可被使用在所有 API 插槽或 Use Camera View API 的 Camera View 插槽中。

UseCameraViewAPI API Next			激活界面 界面蒙层 主界面 主界面类型 界面蒙层类型	Note $\checkmark \Leftrightarrow \wp + \mathbf{x}$ Note $\checkmark \Leftrightarrow \wp + \mathbf{x}$ TestLMG $\checkmark \leftrightarrow \wp + \mathbf{x}$ TAR455 \checkmark TAR455 \checkmark
			加载外面 主界面层级 界面蒙层层级	
			型 of roots 搜索详情 ∡ + TA API	ہ ا
			Camera View Change Pawn	None ₹
▲ * TA LOGIC 按钮唯一标识				
◢ API调用队列	1 数组元素	+ @ >		Camera View
0	Q →) t	Ť	• 私知 • • Change Pawn □
自定义任务队列 场景标签显示状态设置	0 数组元素 0 貼图元素	+ @		

● 打点工具

🇤 MarkEditorMode	🗮 内容浏览器	*	放置actor			
选择对应的编辑模式进	:行使用,场景中点:	由鼠标左键可快	捷关闭			
			打点工具▼			
▲ * TA 打点工具						
场景标注名称						
标签名称		0 数组元素	+ 1	Ì		
标签类		None 🔻 🔶	κ + α			
API调用队列		0 数组元素	+	D		
Task调用队列		0 数组元素	+ 1	Ď		

 插件工具栏提供快速打点工具,指定标注名称、标签名称、标签类后,可使用左 键在场景中点击,在点击位置将生成一个指定标签类的 actor。并将标注名称作为 actor 名称,标签名称作为 actor tag (便于使用 show mark api)。

	F43				
🐠 MarkEditorMode	🧱 内容浏览器	🎸 放置	actor		
选择对应的编辑模式进行	行使用,场景中点。	击鼠标左键可快捷关i	闭		
		+7.5	TRe		
		11出	TĂ		
a " IA II 杰工兵					
场景标注名称		监控mark	t l		
▲ 标签名称		1 数组元素	+ 🗇 🤉		
0		安防	to .		
标签类		BP_PopUpMark -	+ Q +	x s	
API调用队列		0 数组元素	+ 🖻		
Task调用队列		0 数组元素	+ 🗇		

- ▶ 标签类将可以指定所有继承自"KitMarkActor"类的蓝图类,故该功能不仅限与标签
- 打点,可根据开发需要灵活使用。

● BIM 数据简化工具

 该工具通过检测相同模型并替换为同一资源,并从简化后的资源进行实例化,原 多个相同模型资源将合并到同一实例中,同时将保留原模型中的元数据 (metadata),此工具将可大幅优化 BIM 机电设备性能,同时实现构件点击获 取元数据等功能。模型进行实例化后,应配合相关实例 API、绑定实例监听事 件、实例 UUID 等使用。



● 在"MarkEditorMode"中找到"BIM 数据简化工具"

ł	MarkEditorMode模:	t∨ ¶-~	•(* ~	" " ~		▶
	MarkEditorMode 🗙 🗣 放	置Actor				
	选择对应的编辑模式进行使用,	,场景中点击鼠标5 BIM数据简化工具 ~	2键可快损	送闭		
	BIMTool					
	一键导入 Data Smith并优化					
	Datasmith文件路径	C:/Users/Char	lesvane/[)esktop/测试	薮	
	Uasset文件目录	/Game/BIMData	/DataTest			
	Datasmith导入选项					
	实例默认选项					
	BIMTool 分步					
	BIMTool 分步 1导入Data Smith 2替换	重复源 3BIM	I模型实例	ſĽ		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data	重复源 3BIM	模型实例	化		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors	重复源 3BI№ 0 数组元素	模型实例 ⊕	化 立		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes	重复源 3BIM 0数组元素 0数组元素	模型实例 ① ①	化 亡 亡		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes All Mesh Components	 重复源 3BIM 0 数组元素 0 数组元素 0 数组元素 	模型实例	៥ បិ បិ បិ		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes All Mesh Components BIMTool 拓展功能	 重复源 3BIN 0 数组元素 0 数组元素 0 数组元素 	·模型实例 ④ ④ ④	ห บิ บิ บิ		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes All Mesh Components BIMTool 拓展功能 删除选中 Actor下所有子项	 重复源 3BIM 0 数组元素 0 数组元素 0 数组元素 0 数组元素 5 数组元素 5 数40元素 	模型实例	化 立 立		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes All Mesh Components BIMTool 拓展功能 删除选中 Actor下所有子项 选中资源模型实例化	 重复源 3BIM 0 数组元素 0 数组元素 0 数组元素 0 数组元素 3 数组元素 3 数组元素 	模型实例 ④ ④ ④ ● ■ 重复源 	化 立 立		
	BIMTool 分步 1导入Data Smith 2替换 Temp Data Imported Actors Imported Meshes All Mesh Components BIMTool 拓展功能 删除选中 Actor下所有子项 选中资源模型实例化 默认	 重复源 3BIN 0 数组元素 0 数组元素 0 数组元素 5 数组元素 5 世 资源替換 	 様型实例 ① ① ① ① ① 重复源 	化 立 立		

● 在"Datasmith 文件路径"中填入需要导入并优化的 Datasmith 文件,右侧省略号 可在文件夹中选取 Datasmith文件路径 C:/Users/Charlesvane/Desktop/测试数 ...

/DataTest

 在"Uasset 文件目录中"中填入 datasmith 导入后文件生成文件夹,如填入文件路 径不存在,则会自动生成

set文件目录	/Game/BIMData

Uas

在"Datasmith 导入选项"与"实例默认选项"中可设置文件导入设置与实例生成设置,其中实例选项中可设置实例 Actor 最小渲染距离与最大渲染距离。

▼ Datasmith导入选项		¢
几何体	_	
材质和纹理	Y	
光源		
摄像机		
动画		
▼ 静态网格体选项		
最小光照贴图分辨率	64 🗸	
最大光照貼图分辨率	512 🗸	
生成光照貼图UV		÷
▼ 实例默认选项		
Min Draw Distance	0.0	
Max Draw Distance	0.0	

 完成以上设置后,点击"一键导入 Datasmith 并优化"按钮,工具将自动对 datasmith 文件进行优化并在场景中生成实例模型。

MarkEditorMode × 🗣 放置	Actor		
选择对应的编辑模式进行使用,	场景中点击鼠标左键可快捷关闭		
0	BIM数据简化工具 🖌		
F BIMTool			
一键导入 Data Smith并优化			
Datasmith文件路径	C:/Users/Charlesvane/Desktop/测试数]	
Uasset文件目录	/Game/BIMData/DataTest		
▶ Datasmith导入选项			*

● 以上为全自动化流程,如有多个 datasmith 文件需进行共同优化,可使用工具下 方"BIMTool 拓展功能"进行手动优化。

•	BIMTool 拓展功能		
	删除选中 Actor下所有子项	选中资源替换重复源	
	选中资源模型实例化		

● 可在资产库中多选文件夹,并进行类型过滤,选中需要进行优化的模型资产。

▼ PJ_PSCIM	Q	滤波器	₹~	Q搜索	燕子湖-会	展区域-N				展区域-M		~	
▼ ► YanZiHu ■ Build ▼ ► Leasting to		天卡 静态网格体	で HW様二	t HW.	۵ ZJKJ	zJKJ.	L ZJKJ_	ZJKJ	T ZJKJ	₫ ZJKJ	z ZJKJ	T ZJKJ	
■ LocationAc ● 燕子湖-会馬 ■ Geometri ● 燕子湖-会馬	tor 民区域-MEP-1F-Terminal es 民区域-MEP-2F-Terminal		上時失二	14.	<u>+</u>	<u>ж</u> . П	<u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u></u>	<u>ж.</u> д	<u>¥_</u> . <u></u>	<u>+</u>	<u>¥</u>	<u>¥-</u>	
im Geometri ▶ im 燕子湖-会服 ▼ 陸 燕子湖-会服	es 禹区域-MEP-3F-Terminal 禹区域-MEP-B-Terminal 		ZJKJ_ 2_75	ZJКJ_ ∭_75	ZJKJ 火槍	ZJKJ 火栓-	ZJKJ_ 火栓_	ZJKJ 火栓_	ZJKJ_ 火槍_	ZJKJ_ 火栓。	ZJKJ 电箱_1	ZJKJ 电箱_1	
▶ 合集	⊕ Q		161 项(1	61 顶被设	(中)								

- 需先使用"选中资源替换重复源"对资产进行优化
- 再使用"选中资源模型实例化"对选中资产进行实例化,实例化前需确保 datasmith 模型处于场景中,如不在场景中,将导致实例化失败,可如下图将 datasmtih 场 景文件拖入场景中。



- 完成实例化生成后,再使用"删除选中 Actor 下所有子项"对 datasmtih 原模型进行快速删除。
- 完成 BIM 数据优化后,可通过"On Click Instance Listenner"对鼠标与实例模型之间的交互进行监听。

● ●件开始运行 ■	7 On Click Instance Listenner
	D D
	Callback
◆ 自定义事件 自定义事件	
D	
υυο 💁	
Instance Actor 📀	
Instance Index 📀	
Click Key 🕥	
Event Type 🕥	
Meta Data Struct 💿	

- 其中"UUID"可获取实例 Actor 内每个索引模型的 UUID, "Instance Actor"可获取 实例 Actor, "Instance Index"可获取当前与鼠标交互的模型在实例 Actor 中的索 引, "Event Type"可对鼠标交互事件进行区分, "Meta Data Struct"可获取与鼠标 交互的模型元数据、世界坐标。
- 由于实例模型的特殊性,对选中模型的描边无法使用"Set component outline"API,需使用"Set instance outline"对选中实例索引进行描边。实例描边已 对 Nanite 模型进行兼容,可以 Nanite 模型进行描边。



● 描边颜色可在插件资产中搜索"PP_Outline_inst"材质进行设置。

○ 内容浏览器 ×										
+ 添加 👌 导入 📲 保存所有 🕒 🖻 🔺	네 > 리의	🛚 🕻 🕻 Plugins 💙	TECHNOLO	GY ART内容 >	ArtContent	> AnalysisT	ools			尊 设置
▼ PJ_PSCIM	Q	滤波器	₹× Q	異察 AnalysisTo					•	
▶ man Synthesis and DSP Effects内容 ▶ man Synthesis and DSP EffectsC++类 ▶ man Take Recorder内容 ▶ man Take RecorderC++类 ▶ man Take RecorderC++类									X=1 Y=5	
◆ TECHNOLOGY ART内容 ◆ ArtContent AnalysisTools							PP_Outline_ Inst	RT_Skyline2D	ShadowMap Paramters	ShadowMap Texture
 ▶ ■ DevResource ▶ ■ FBXImporter ▶ ■ FBXImporter ▶ ■ Labels ▶ ■ TrafficSystem 				Ć	C					
▶ Buccele ▶ Buccele ▶ BuccenseForPak Slate ▶ TALibrary	l			VisualField_ Mat	VisualField_ Mat_Inst					
▶ 合集	⊕ Q		11 项(1 项被;	责中)						

● 实例 UUID 同样可在管理器中使用,在场景中可通过点击实例 Actor 后右键该实 例 Actor 中的

单个模型,快速复制该模型的 UUID。

Scene•		ProjectUILogicTool• ×			8	HERE LIP
a 🛱					and the second	
拉件图表	场景点击事件图表 ×			Q 提索		
				w New Widget		
				Create New Widget		
					Mask	6
				Apply Nodes		
				请试模式		
		UUID Logi: Node		放活开面	None 🗸 🕒 🗊 🔿 🗙	
		点击 🔿			Mask 🗸 🗲 📭 🕀 🗙	6
		単行 〇			15m 🗸 🕒 🕅 🖌	45
		停止悬停 🔿			TA.菜単类型 マ	
					TA,菜単类型 ❤	
					None 🗸 🕒 🕅 None	
Bo destances of						
∎p №9918382 ×						
十 漆如 97 台 1	🛢 保存所有 💿 🕘 🖿 All > 内容 >		- ⁴ 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	Q 88		
▼ PJ_PSCIM	Q 〒 ✓ Q 現象内容			w APILegic Node		
T 📂 AJI					KitinatanceMeahActor_421.0	5
▼ ► 内容 API						
Blueprints						
Developers						
► Maps ► ProjectData						
UMG Volumetric Cloud						

● 自定义编辑器功能拓展

类似 BIM 数据简化工具的做法:



创建该类型蓝图 BaseToolObject:

u	Pick Parent Class	×
▼ COMMON		
🧕 Actor	An Actor is an object that can be placed or spawned in the world.	0
•	A Pawn is an actor that can be 'possessed' and receive input	



创建以下测试方法:

(11)	File	Edit	Asset	View	Debug	g Window	Tools	Help
		BP_Test	ToolObje	ct*	×			
🕐 Com	pile	:	Save	Di Bro	wse	■ 🖉 Diff 🗸	🗩 Find	d •)
🛤 My Bl	ueprin	nt ×						
+ Add	0	Search						\$
T GRAPHS	5							Ð
💦 Eve	ntGra	ph						
	ONS (2	OVERRID#	ABLE)					Ð
f tes	сті							
f tes	ST2							
∫ TES	ST3							
MACRO	s							€
VARIAB	LES							€
EVENT (DISPAT	CHERS						Ð
LOCAL	VARIAB	LES (TEST						Ð
🗾 Detai	ls	×						
Q Sea								≣‡
🐨 Graph								
Descrip	otion							
Catego	ry			Defau	t 🗍	~		
Keywor	ds							
Compa	ct Node	e Title						
Access	Specif	ïer		Public		*		
Pure			_			1 2		
Call In I	Editor			2				
Advanc	ed					-		
Const								
Exec								
Dorread	sare							
Deprec	ation M							
✓ Inputs								Ð
Please	press t	he + icon a	above to a	idd parame	ters			0
🗢 Output:	5 .							Ð
Please	press t	he + icon a	above to a	idd parame	ters			
	TEST			f P	rint Strin	ig		
				-	التناييا		D	
				0 V	Vorld Con	text Object		
				01	n String [Test1		

其中有一点比较重要的是重写这个鼠标点击事件 (如果需要的话)

Aug ocaron	
GRAPHS	\odot
📭 EventGraph	
FUNCTIONS (2 OVERRIDABLE)	Override 🗸 🕣
f testi	
f test2	EDMode Mouse Click Base Tool Object
f test3	Run Editor Utility Object
MACROS	•
VARIABLES	\odot

Development Only



其中 KeyName 是: 捕获点击事件是左右键的区分 分别是:

左键/右键 || LeftMouseButton/RightMouseButton

ClickLoc 是点击的点在场景中的坐标

reture 的 bool 值 控制的是是否阻止继续向下传递调用: true 时则只执行自定义的部分

false 时会调用原先的左键选择功能和右键关闭工具功能和右键菜单弹出功能。 开发完成后在 config 中进行配置:

🔱 🗣 Project Settings 🛛 ×			
All Settings	Q Search		
Project		0 Array elementa 💿 🖞	
Description		projectid	
Comentaria			
Mans & Moder		4.26.1	
Movies			
Packaging			
Supported Platforms	联网环境	Develop	
Target Hardware		http://127.0.0.1/	
-		http://127.0.0.1/	
Game		CharlesAPI	
Asset Manager	联用服务器(p		
Asset Tools			
Dev Kits Config			
Engine			
Al System Animation	仅偏缓模式下的标记材质	Nere C Transformer C Transform	
Animation Modifiers			
Audio			
Chaos Solver	👳 • CONFIG UILogic		
Collision		NewULogicTool2	
Console	方面在父		
Control Rig			
Cooker			
Crowd Manager			
Data Driven CVars	♥ • CONFIG EDMode		
Debug Camera Controller		1 Map elements 🕑 🖞	
Gameplay Debugger	жівжута	BP_TestToolObject 🛩 🕑 🍺 🗙 🗸	

然后需要重启编辑器,可以看到自定义工具被插入到工具选项中了:

	📥 MainLevel			
	MarkEditorMode Me	ode 🗸 🛛 🤹	• " ~ ""	
MarkEo	ditorMode ×			
选择对	应的编辑模式进行使用,	场景中点击鼠标左键	可快捷关闭	
		关闭 🗸		
		关闭		
		打点工具		
		视角工具		
		样条线绘制工具		
		AI行人工具		
		API生成工具		
		合并工具	1	
		测试自定义工具		
		BIM数据简化工具		
开启工具	测试,可以看到调用	用编辑器方法成功	了,并且鼠标事	事件也顺利打印:
MarkEditorM	ode ×	8	Perspective 🕑 Lit	Show
选择对应的约	编辑模式进行使用,场景中点击鼠标左键 测试自定义工具 >>	 打快捷关闭	REFLECTION CAPTURES NEED	D TO BE REBUILT (1 unbuilt)

淜试自定义工具 ✔	Test3 Unit ble AliGoreen Massages to appress Test2
Default TEST1 TEST2 TEST3 Call an event on the selected object(s) TEST3	Test1 X=5963.379 Y=-2570.978 Z=1238.901 LeftMouseButton X=5963.379 Y=-2570.978 Z=1238.901 RightMouseButton

接下来就可以随时开始进行编辑器工具的开发拓展了,直接使用蓝图即可无需 C++

● 图片序列功能拓展

● 拓展功能概述

播放图片序列文件时,通常需要在后半段进行循环展示播放,其中不循环的前 半段作为图片序列的入场动画,插件将针对该需求进行拓展。

• 图片序列功能使用

● 在项目设置中添加序列 UMG 模板,使用插件自带模板"TP_FlipBook"

(11) X# 1818 RD IA	桥助	
● 项目设置	× 🚣 KingJDE	
所有设置	(Q. 限票	
项目	- 游戏 - Dev Kits Config	
打包		导出 导入
地图和模式	🔓 这些设置被保存在DefaultDevKitsSetting ini中,它当前可写入。	
加度	▼ IA	
<u>編33</u> 日标硬件	File Back Towardship Path Old Back	TP_flipBook V
影片		
支持平台	Template UMGs	○私間元素 ⊙ 豆



● 导入图片序列文件,并将纹理组改为 UI



- 选择需要创建为序列的图片文件
- 在右键菜单中找到"Create Sprite Player

<u>hann</u>	la anti	Same.	Same .				
区位图 00005	区位图 00006	区位图 00007	区位图 00008	区位1 0000	۲	创建材质	
100 Table	W	W		1885	82	转换为虚拟纹理	
				8	尿	创建Slate笔刷	
ALCONE TACANT	17.45 M	2000	200 (000) 17 (4 m)	Real Provide State	22	创建纹理阵列	
00010	00011	00012	00013	000	8	Create Sprite Player	
1.2	17 28	17 30	1		×	Spriteimte	,
le al		L.A.	le al				
区位图	区位图	区位图	区位图	区位		重新导入	
00015	00016	00017	81000	0001		打开源位置	
19	17.20	1	17.38	1	R	在外部编辑器中打开	
	E. M	6.1	1				
区位图_	区位图	区位图	区位图	区位		编辑	
00020	08821	00022	00023	0002		重命名	F2.
	1				5	复制	CTRL+D
Ξ×						保存	CTRL+S
区位图	区位图				ŵ	删除	DELETE
00000_					J.C	资产操作	>
					PH	资产本地化	>
					ã,	在文件夹视图中显示	CTRL+B
					Ϊø	在浏览器中显示	
					F .;	复制引用	
					F .;	复制文件路径	
					:•§	引用查看器	ALT+SHIFT+R
					10	尺寸贴图	ALT+SHIFT+M

• 创建文件完成后可以双击进入文件进行循环帧、帧率等设置



 "Is Autoplay"可设置是否自动播放 "Is Loop"可设置是否使用循环 "Loop Start Key Frame"可设置开始循环帧 "Play Rate"可设置播放速度 "Start Key Frame"可设置播放开始帧 "Frames Per Second"可设置每秒播放帧数

● 右键 Texture Player 文件,点击"Create Texture UMG Player"即可创建序列控件 UMG

控件带	50	拉伊爾	"	1
-	TEXUR	E PLAYER 操作		
	8	Create Texture UMG Player		
17.65.00	進用			
00000		编辑		
3051200	εĮ	重命名		F2
80000	۹Į)	重命名		



● 该 UMG 可以在其他控件 UMG 中使用或蓝中加载



● 创建的序列 UMG 拥有"Play"函数,可以在非自动播放的情况下手动播放序列



- 其他功能拓展
- 真实坐标系转换到 UE 坐标系
- 插件支持快速转换 WGS84 坐标系、GCJ02 坐标系(火星坐标系)转换到 UE 世 界坐标系

0	
O Longitude	
📀 Latitude 🗌	
Or Scene Point Loc X 0.0 √ 0.0 Z 0.0 0.0	
f GCJ02to World Position	f WGS84to World Position
f GCJ02to World Position	J WGS84to World Position O Longitude World Position

● 先通过"Config GISConverter"节点对 UE 场景参考点进行标定,如在 UE 场景中坐标(0,0,0)所对应真实世界中的(114°E,22.5°N)。如为西经/南纬,值为负。



● 根据需求使用 GCJ02 或 WGS84 的转换节点获得 UE 世界坐标

◆ 自定义事件 自定义事件		〕 〕 〕
	Congrude 114 Congrude 114	h String 仅质开发
	● Laittude [2] 333 F WGS84to World Position ● Longitude [115222] World Position ● Laittude [2] 333	

TechnologyArt-UILogicTool

● 基本概述

UILogicTool(以下简称 UI 管理器),以模块链接的方式快速创建各个 UMG(UI 界面)之间的交互逻辑,可使用户以无代码的方式快速完成交互设计。

● UI 管理器创建与使用

● 创建 UI 管理器并为项目配置

注意:根据首节 License 文档中的配置步骤操作后,项目使用的 UI 管理器将在创建时的指定路径生成,以下为未创建时情况。

● 在资产管理器中右键空白处,在弹出菜单中点击"UILogicTool"项



<u>影片</u> 支持平台 <u>GameplayTags</u>		• NER.
游戏		
一部产管理器	▲版本控制	
Dev Kits Config		projectid
この整		1.0
	UEVersion	4.22.1
层版LOD 局前局將依	Debug	新建资产
导航系统	₄ + CONFIG Net Work	
地形		Develop 当前资产
调试摄像机控制器		http://127.00 265
动画		http://127.0.0 all
关于序列	联网环境 	CharlesAPI atw.
結瘍器	联网版务器Ip	
教程	000mm	NewUlLogicTool
投影台		
<u>坦波回收</u> 注法	⊿ + CONFIG Mark	
群集管理器	仅编辑模式下的标记材质	
输入		
数据驱动CVars	4 * CONFIG FBX	
网络		
物理		
選出 2010年47月1日日	∡ * CONFIG UILogic	
<u>過宋里和(卒宅)</u> 		± the second se
<u></u>		Note $ \phi \rho $

- 使用 UI 管理器
- 双击资产打开 UI 管理器

				e – a ×
宿放1:1	技术评估			~ی 🏢 🛛
	⊿ New Widget			
	Create New Widget			
	▲ Tool Function			
	Apply Nodes			
	a • CONFIG UITools			
		None 🖛 🗭 🤊	+ ×	
	齐函兼居	None + P ·	+ ×	
		None 👻 🔶 🔎	+ ×	
	主界面类型	TA_菜单类型 ▼		
	528622 10185			
	土市市民的			
	乔面蒙屈屈吸	0 2		
	⊿ UITools			

 为接下来需要创建的 UMG 控件填写名称,点击 Create New Widget 创建一个 UI 管理器可控的 UMG 控件资产

u *****	10 20日四日	V O Newdlasgistiool*	× All and the					N
文件 编辑 资产 窗口 帮助								
× 184								
144 ME								
					挂滚评情			ہ 🏾 🍳
					∡ New Widget			
					Create New Widget 2			
					New Widget Name	TestUMG	1	
					⊿ Tool Function			
					Apply Nodes			
					A + CONFIG UITools			
						None 🖛 🔶 1	• + x	
					件面做层	None 🕶 🔶 3) + x	
						None 💌 🔶 1	• + ×	
						TA_菜单类型	-	
					界田蒙尼夷型	TA_就单类型	-	
					016197-00	None 🗸 🗧	, + x	
					17660 889220	0		
							_	
					⊿ UIToola			

注意:不建议直接使用 UE4 原生用户控件类,如仍需要使用原生用户控件类需知悉 以下可能出现的问题。 1.UI 管理器无法管理该类 UMG

2.部分 API 无法控制该类 UMG

● 新的 UMG 控件将创建在内容资产下的 UMG 文件夹,点击保存所有保存该资产



● 回到 UI 管理器界面中,在主菜单项下拉菜单中搜索新创建的 UMG 控件,并选 择指定

			r ×
文件 编辑 资产 窗口 帮助			
※ 工具栏 ×			
缩放1:1	搜索详情 New Muger Name	Testowig	
	∡ Tool Function		
	Apply Nodes		
	▲ * CONFIG UITools		
	激活界面	Noneマ キ の 🕇	×
	界面蒙层	None▼ ← ク +	×
	主界面	None 🗸 🔶 户 🕇	×
	主界面类型	test	X
	界面蒙层类型	O None	
	加載界面	2 10	Test UMG ● 视图选项▼
	主界面层级	0	
	界面蒙层层级	0	
	LIITool Man		4 10
	401-		
	APIS		T U
	r		

- 指定的 UMG 将作为该项目启动时默认加载界面(入口界面),其他界面交互皆 基于该界面
- 右键左侧空白处,在弹出菜单中选择"CreateNewUILogicNode"项,创建一个 UI 管理节点。

U O New Class of out + +		
文件 編輯 资产 留口 帮助 10 丁目#		
保存 創業		
	缩放1:1 搜索详情	
	⊿ New Widget	
	Create New Widget	
		TestUMG
	∡ Tool Function	
	Apply Nodes	
	▲ + CONFIG UITools	
		None + A + X
	界面蒙层	None▼ ← ມ + ×
les	主府面	TestUMG + P + X P
A Generic Graph Node	主界面类型	TA_菜单类型 ▼
CreateNewAPILogicNode CreateNewUILogicNode	界面蒙混类型	TA、菜单类型
Add mode here		Kone + D + x
	工作の形成	
	⊿ UITools	
	Ultooi Map	○ M田元素 + 四
		0版的元素 + 10

● 在右侧面板 UlInstance 栏下拉菜单中搜索指定的主菜单,并选择指定



▶ 将 Add Type 指定为"TA_菜单类型",该类型将会使主菜单界面保持显示。

⊿ UEd Node UILogic Tool Node	
Ulinstance	TestUMG ← ♀ ⊐
Zorder	0
Add Type	TA_菜单类型 ▼ つ
	自定义类型(没有下一个) TA_菜单类型 TA_隐藏上一个UI TA_叠加旧的UI上面 TA_单个显示类型

● 点击 Apply Nodes 和保存按钮,将该改动应用

11 OWHOLDDIGG		★ = 0 ×
Y IN:		
FI DE		
缩构1-1	搜索详情	• ≣ ۹
	⊿ New Widget	
	Create New Widget	
		TextUMG
	a Tool Function	
	Apply Nodes	
	▲ • CONFIG UITools	
		None + P + X
	界面兼层	None + p + x
		Testimo + p + x v
	王界运员型	TA. 数单类型 →
	1088822	
	主界面层级	0 0
	界面就是意识	0 2
	⊿ UITools	
O Page		
	波索详情	۵ 📖 🛛
		Tentana 🗸
		+ Q +
		•
		TA_菜单类型 ▼ 5

注意: 在 UI 管理器中,完成改动后都应该点击 Apply Nodes 与保存按钮,以避免相 关改动未应用

● 双击节点进入主菜单控件界面,为该界面添加数个 Kit Base Button 或 Kit Check Group Item,添加完成后点击左上角"编译"并"保存"。



● 回到 UI 管理器中,右键节点并选择"刷新节点"。

引脚操作			
F Node Actions			
🚽 🗎 🙀			
🤙 复制			
◎ 粘贴			
刷新节点			
断开节点连接		刷新节点	
Apply Nodes	空格键		
Open UI	回车键		

此时 UI 节点将显示控件内按钮的输出接口,从接口中拉出箭头并创建下一步节点,由此完成交互逻辑。



注意:当前版本中,UI 管理器的 UI 节点仅识别 Kit Base Button、Kit Check Group Item 两种类型组件。

● 多级 UI 嵌套

- 当有多层级子菜单嵌套时,应当尽量遵守"切换时回退上一级"的准测,下列步骤将 演示如何遵守该准测。当熟悉 UI 管理器逻辑后,可根据开发需求自行决定是否 需要回退,并灵活设置 Add Type。
- 在 UI 管理器中创建 3 个子页面 UMG,为其中任意一个子页面控件添加任意数量 按钮。

搜索详情	ه 🏾 ۵
a New Widget	
Create New Widget	

New Widget Name		子页面1		
247 Max 37.4 MB 40.4 ML MAX → 1.47 M2 2.5 ML MAX 41.4 MB → 1.51 M → 1.5	は対象- 28番 の ⁹ 第放1:1) 200 500 400 508 NO	. 190 90 900 1000 1195 13	
6 केंद्र - अस -		F		
10 10				
				DUIGHO

● 在 UI 管理器中添加这 3 个子页面(创建节点后,右键刷新可获取到该 UMG 内的按钮),并在 3 个子页面节点右侧面板中,设置 Add Type 为"TA_叠加在 旧的 UI 上"。

TestUMG	一 子页面1	
Page KitCheckGroupItem 99	◆ Page KitCheckGroupItem_64	
KitChaoloCroupItam 151	KitCheckGroupItem_94	
KitcheckGroupitem_151	KitCheckGroupItem_152	
KitCheckGroupItem_262	KitCheckGroupItem_182 D	
	25.00	
	O Page	
	子页面3	
	O Page	
文件 NAM 我们 彩云 和地		0.00000
× 194		CO REALES
Proti		CO ANALON
	15812) Jacoba Jacoba	£ 1000 €
	A survey and the survey of the	ρ <u></u>
р так ал ал О Принимание О	التقالية الاستعمال الالا الاستعمال المالم المالمالمالمالمالمالمالمالمالمالمالمالما	ο
р Гон ал 20	1980 1980 1980 1980 1987 1980 1987 1980 1987 1980 1988 1980 1988 1980 1988 1980 1988 1980 1988 1980 1988 1980	<u>0</u>
2 ° 20° 20 ° 20 0 ° 24 ° 20 0 ° 24 ° 2000 operation • Contraction • Contrac	Computer Computer Computer Original (Computer) (Co	£)
Profession Profe		<u>β</u>
Provide and the second se		β
A The second sec		
ротото по по п	721	
ротото по по п	TTE ************************************	
ротот по по п		
		β

由于 3 个子页面均使用叠加类型,此处应在主菜单与子页面之间添加"跳转界面"
 类型 API,并选择跳回"TestUMG"。此处如不遵守"切换时回退上一级"的原则,3
 个子页面将按照点击顺序叠加在主菜单之上。



 依次类推继续增加下一级内部页面,依然将所有内部页面设置为叠加类型,此时 内部页面的上一级为"子页面 1","跳转界面"应跳转至"子页面 1"



● 灵活拓展

 某些开发情况下,可灵活使用 Add Type,如制作一级子页面时,可不进行回退, 仅将所有子页面设置为"单个显示"类型即可实现相同的功能逻辑。



注意: 当未遵循"切换时回退上一级"时,默认 check 的勾选组将会保持当前勾选状态,故当离开有默认勾选功能的页面时,应当在该页面的 Onhide 中手动将勾选组的 勾选状态恢复为"-1"。



● 场景点击事件图表的使用

● 基本概述

在 UILogic 中,除了可以对 UI 控件交互进行管理外,还支持对场景模型、POI 的交互进行管理。在"场景点击事件图表"中,可以通过使用 UUID 对指定场 景模型、POI 绑定 API 事件,同时支持绑定到点击、悬停、未悬停事件。

● 获取模型 UUID

点击场景中任意 Actor,点击上方菜单"复制 UUID"项,在下拉菜单中,将自动识别该 Actor 的 Actor ID 及所有模型(static mesh)组件 ID。点选所需的 ID,将自动复制到粘贴板。

Ľ	文字 新聞 第二 工具 特徴 送昇 Actor 特徴 Actor 特徴 PropertitiongleTool	TATOPHES _	σ×
1 6 SEA	Gassar→ Gマ・ロマ 当マ トロム 日 展7日マ ム err × 取りAsias		¢as.√
Q	○ A ♥ ○ ≤ *: A □ ≠ THATMENT ART	4 - 3069 - A5 0 - A Core (2013) 18-9 - Choix 317	D-ERI
		A Gyana kini pilogo B hathamania S kainy A Wana Kina A Wana Kina	
<u>т</u> ±		1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	
		E the Specific was Related by Specific and S	
<u>.</u>		Z des × © delike ja	
2		(# Cole + HEL + (# One (ER))	
#0 A10			лиятик.
	The second secon		
L	线性阻尼	0.01	
l	角阻尼	(11) 复制成功	
	启用重力		

● 获取 Widget UUID



- 该事件用于监听鼠标检测所击中的组件
- UUID 在蓝图中的应用

Scener Maria I Scener

	o <u>i</u>			
Е				• • • • • • •
	x ve		Tim Com	0
		+	axuu –C : `	
				+
Mark Mesh (markMesh)			在C++	中编辑
			11 11 13	17月7日9月7日
APIComp (APIComp)			在C++	中编辑
○ 搜索				يدر 👞
	500 Str.	56 Z		M 24
四府 LOD 东坝 物理	追 栄	网伯		
▼ 受抉	100	lee	100	
	Toot		0.0	
	110	11.0	0.0	
	U I.O	1.0	1.0	
元素 0		Widget3DPas	ssThrough_Ma I ✔	ski 🕤
▼ 用户界面				
空间	屏幕	~		
计时规则	現实时间	~		
控件类	BP_PopU	pTip 🗸 🗲	×⊙⊲	
▶ 绘制大小	500	50	D	
手动重绘制				
重绘制时间				
以所需大小绘制				
▶ 枢轴 	0.5	0.5		
儿何体模式	440			
回杜怀强形用度	180.0			
TICK 侯式	已启用	~		
100 187 AHI (CLAR) A				

点击场景中任意 Actor,点击上方菜单"复制 UUID"项,在下拉菜单中,将自动识别该 Actor 的 Actor ID 及所有父类继承自"kit base widget UMG"的 Widget 组件 ID。点选所需的 ID,将自动复制到粘贴板。

04******

	Callback
◆ 自定义事件_0 自定义事件	
D	
UUD 🔿	
Component 🔿	
Click Key 🔿	
Event Type 🔿	

● 可通过 UUID 获取对应 Static Mesh 组件

🖵 Get Mes	h Component by UUID	
D		D
🔿 UUID 🦳	Return Value	0

● 在图表中新建 UUID 消息节点

● 打开 UILogic,在选项卡中找到"场景点击事件图表"。

💾 保存 🏹 🔅	刘览				
None	×				
控件图表		场景点击事件图表 ×	¢		
				الا الح الح الح ال	

● 在图表中右键菜单中点击"新建 UUID 消息节点"



● UUID 消息节点的参数中,将会自动粘贴最后一个复制的 UUID

		Q搜索		■
UUID Logic N	lode	V APILogic Node		
	点击 🕒	Node UUID	StaticMeshActor_1.StaticMeshComponent0	
息停 〇	Vit Base Node			
停止	急停 〇	Custom Node Name		

● 在所需的交互事件后链接 API 或 UI 控件即可



TechnologyArtAPIs

● 基本概述

基于常用基础功能进行封装,可在全局及插件拓展的管理器中快速调用, 该类封装功能称之为"API"(以下简称 API)。API 拥有基本参数设置,也可对其封装功 能进行二次拓展。



(API 结构图)

- API 创建与使用
- API 编辑器创建 API 文件
- 点击上方菜单"模式"的下拉箭头,勾选"MakeEditorMode"项,在弹出的面板中点击"API 生成工具"按钮开启 API 编辑器

MarkEditorMode ×	🗣 放置Actor	〕 内容浏览器
选择对应的编辑模式进	行使用,场景中点击	鼠标左键可快捷关闭 API生成工具 ❤
🗢 * TA API		
APIName		
APICIass		请选择类型 🗸
当前API实例		None € ₽
Create New API		

● 为创建的 API 填写一个名称,点击 APIClass 项的下拉箭头为该 API 选择一个 需要的类型,完成选择后点击"Create New API"按钮

MarkEditorMode ×	🗣 放置Actor	6 内容浏览器		
选择对应的编辑模式进	[行使用,场景中点击]	鼠标左键可快捷关闭 API生成工具 ❤		
🔻 * TA API	_			
APIName	Ľ	showmark		6
APIClass	ſ	控制标签显隐 🖌	2	с э
当前API实例 Create New API	3	None € ₯		

● 此时在文件夹中将会自动创建一个新的 API 文件,并自动将该 API 填入"当前 API 实例"插槽中。

MarkEditorMode 🗙 🗣 放置Actor	▶ 内容浏览器	
选择对应的编辑模式讲行使用 场墨中台	5. 书题标左键可快捷关闭	
	API生成工具 🗸	
▼ * TA API		
APIName	showmark	€
APICIass	控制标签显隐 🗸	\$
半前ADI守街	showmark1 ~	5
	a e 👱	
Create New API		
Mark Nama		
Mark Name		
W Kit Base API		
APIType	控制标签量隐 🗸	
🐨 Execute Json Msg		
API_Json		

● 此时在 API 编辑器中出现该 API 可修改参数,按需调整即可完成 API 创建

MarkEditorMode ×	🗣 放置Actor	办 内容浏览器	
选择对应的编辑模式进行	亏使用,场景中点击 鼠 材	示左键可快捷关闭 API生成工具 ❤	
🔻 * TA API			
APIName	s	howmark	ۍ ۲
APIClass	15	2制标签显隐 🖌	¢
当前API实例	4	showmark1 ✓ € ₽	¢
Create New API			

▼ * TA API		
Mark Name	消防	¢
State	~	¢
🔻 Kit Base API		
APIType	控制标签显陶 🖌 🖌	
🔻 Execute Json Msg		
API_Json		

● 在 UI 管理器中创建 API 文件

● 点击进入项目使用的 UI 管理器中,在空白处单击右键,在弹出菜单中选择 "CreateNewAPILogicNode",创建一个空白的 API 调用节点。

				r = 0
文件 编辑 87* 11日 年間 12* 工具性				
97 33				
		1 建滚译情		P I
		∡ New Widget		
		Create New Widget		
			TestUMG	1
		Apply Nodes		
			א + a + א	
		件面前屈	א + a + א	
			TestUMG + D + X 5	
		主外面突型	「人菜羊类型」	
		DBBB		
		主界面层级		1
	A Groenic Graph Node	界面蒙屈层极	0 0	
	CreateNewAPILogicNode	↓ UITools		
	Clearner OL Ogithole	UlTool Map		
TestUMG				
Page KitBaseButton_99				
KitBaseButton_151				
KitBaseButton_226				
KitCheckGroupItem_63				
				1

● 在右侧面板中选择一个需要的 APIType,即可创建一个新的 API 文件。



● 通过与 UI 节点的按钮输出链接,可使程序按照链接顺序执行 API 或加载新的界面



● 在 UI 管理器中使用 API

● 点击进入项目使用的 UI 管理器中,在空白处单击右键,在弹出菜单中选择 "CreateNewAPILogicNode",创建一个空白的 API 调用节点。

					😵 – Q
文件 编辑 资产 第日 相助 12 工具栏					
			据宏详情		
			∡ New Widget		
			Create New Widget		
				TestUMG D	
			Apply Nodes		
			∡ • CONFIG UITools		
				None + P + X	
			558G		
			土竹田 中房面並型		
			界面集团类型	TA_菜单类型 ▼	
				х + Q + х	
	吉奈	P	主界面层极	<u> </u>	
	Generic Graph Node Crostablem 3 Bill operation		界面繁层层级	0	
	CreateNewULogicNode		⊿ UITools		
TestUMG					
O Page KitBaseButton_99 D					
KitBaseButton_151 🜔					
KitBaseButton_226 Ď					
KitCheckGroupItern_63 🕥					
			-		
● 左右侧面板"ADI Insta	nco"括樯山北宁雪	更估田白		/ 在	
▼ 1工/口 四 11 八 AFIIISIA	山にゴ田宿T泪化而	マ区川口			

⊿ New Widget	
Create New Widget	
New Widget Name	TestUMG
▲ Tool Function	
Apply Nodes	
▲ • CONFIG UITools	
激活界面	None - + + ×
界面蒙层	None▼ ← ♀ + ×
主界面	TestUMG - + D + X >



● 或选择 API Type 类型进行新建 API

	# • CONFIG UITcole	
		1000 + 0 + x
		1000 + p + x
		Tentes + p + x p
		14 2429 -
	768/C29	14 R#29 -
	231750	
APPeide Dase	+ 5 X/M	
DAN Word		
	a UlTeole	
	C Test MARK BERNARD AND AND AND AND AND AND AND AND AND AN	
	TOLEUMELX ROTE IN DOUGLASS, 191	
	A Test EMB_C4D architesphare_202	
		Magellogradiantification (2011)21122122 + D + X
		TA:单个担保类型▼
	 V3813) Oeckingten, M 	
	• #dill.koohekonsphen_H	-
	7.381 Kitheximpten, IQ	83.022
	 All Dodecompting to 	10.500 ACT
		(1)169月 考以予慎
		388843 128X-100
		· 注意Sequence 教品 · 注意时间
		25%(N-3) 2000 古堂 (24)
	建窑钟情	
	APILopic Node	ANDIA MCM ¹
		《秋秋時》章 故來書的我會An
	Mindana	日本(11)(11)(11)(11)(11)(11)(11)(11)(11)(11
		雪(本氏見) 単寸(カ氏肌)
		THE REPORT OF A

● 在 UMG 控件内使用 API

● 在控件界面中,添加一个 Kit Base Button 和 Kit Check Group Item

11 New Charles Bloeprint 또 또 New Charles Bloeprint	Men Magealayrin.	- 0 ×
	Har Ha (155) Ha (155)	设计器 〉 🚛 图表
Britis Britis # BR * C Rold * C Rold * M Direk Group Into		
TE IND SECON SECON SECON SECON SECON I I I I I I I I I I I I I I I I I I I	の	
◆ 20画 ● 200画 ● 200画 ● 200画 ● 200画	pe arona (B) na menenenenenenenenenenenenenenenenenenen	

点击 Kit Base Button 或 Kit Check Group Item,在右侧细节面板中找到"API 调用队列"项,点击右侧"+"按键,向队列添加一个 API 文件插槽,如需调用多个 API 文件,则添加多个 API 文件插槽。

				N	0.0
xi an yr με αι μι μι δι - μεταγία μι ατ με με με μι μι μι μι ατ με			1	Qit器 〉	
● P28/HE 0	1500 2000				
check × 缩放-3 704x857		KitBaseButton_27		Is Variable	打开KitBaseBut
		接滚译情			0 11 0-
· on wasa		▲插槽(面布面板槽)			
+ Kit Check Group Item		D GIA	積点		
			0.0		
			0.0	3	
			100.0	3	
			30.0		
			X 0.0	Y 0.0	0
		大小則内容 +			
		1. TALOGIC			
		1- TALOBIC			
I Marcine Constant Co		APTIKATIAN	0 IXIIISER		
		目定义性务队列	の数理元第	+ =	
 Niclestroughture_83 Image: A state of the s		结果检查显示状态设置	0 短期元章	.**	
		医动物一种学			
• 设备内容指放1.0		・ 営業館色 +			
末设置设备安全区	······	⊿可访问性			
1280 x 720 (16:9)	DPI缩放0.67 😋	重转可访问默认值			
オカ首 ガ首 ガガ首 ガガ首 ガガ首 ガガ首 ガガ ガガ					
and area β					

▶ 指定 API 文件后,在程序运行中点击该按钮将由上至下执行 API 逻辑。



同样在按钮右侧细节面板中,可快速调用 ShowMarkAPI,无需创建新的 API 文件。点击"+"按键添加一个 Mark 名称(Actor Tag)及状态插槽。在程序运行中点击该按钮将执行该 ShowMarkAPI。

 ▲ * TA LOGIC API调用队列 0 数组元素 + 面 自定义任务队列 0 数组元素 + 面 场景标签显示状态设置 0 贴图元素 + 面 按钮唯一标识 	201061 +	U		
API调用队列 0 数组元素 + 面 自定义任务队列 0 数组元素 + 面 场景标签显示状态设置 0 贴图元素 + 面 按钮唯一标识	▲ * TA LOGIC			
自定义任务队列 0 数组元素 + 面 场景标签显示状态设置 0 贴图元素 + 面 按钮唯一标识	API调用队列	0 数组元素	+ 🖻	
场景标签显示状态设置 0 贴图元素	自定义任务队列	0 数组元素	+ 🖻	
按钮唯一标识	场景标签显示状态设置	0 貼图元素	+ 🖻	
4 4 5.70	按钮唯一标识			
2 7 7 8	⊿外观			

● 在插件拓展的 Actor 中使用 API



在资产管理器右键菜单中选择创建蓝图类。



● 在弹出窗口中搜索"KitBaseActor",并选择创建该类

U	选取父类	×
⊿ 常见类		
Actor	Actor是一种可在世界中放置或动态生成的对象。	0
8 Pawn	Pawn是一种可以被"控制"的Actor,且可以接收来自 Controller的输入。	0
()角色	Character是Pawn的一种子类型,增加了可四处走动的功 能。	0
💢 玩家控制器	PlayerController是Actor的一种子类型,其负责控制玩家所使 用的Pawn。	0
🔄 游戏模式基础	GameModeBase定义了正在进行的游戏、其规则、得分以及 游戏类型的其他方面。	
 Actor组件 	ActorComponent是一种可复用组件,能被附加到任意Actor 上。	0
🔍 场景组件	SceneComponent是一种组件,能够进行场景变换并可被附 加到其他场景组件下。	0
▲ 所有类		
kitbaseactor		×
▲O Object		
Actor		_
2.15(1.15抽进中)	——————————————————————————————————————	416-
5项(1项版选干)		西班
	选择 取;	肖

● 将创建的 Actor 拖拽到场景中



 在世界大纲中选中该 Actor,将在下方细节面板看到与控件界面中相同的 API 插 槽设置。插槽中指定 API 文件后,程序将在点击到该 Actor 下挂载的任意碰撞 后执行。



- 在蓝图中使用 API
- 在全局所有蓝图中均可在右键菜单中找到 Kit API 下的 Caller 项,可直接调用所 有 API

1.4754.07.249.h			
uter the total			
业共同化成大规定			
业共同地区大地化			
1. 监督的时有操作	✔信境关联》		
提索	0		
洗择组件来查看可用事件和函数			
4 • Kit API			
J Delay Time API			
Mouny Controller Model API			
f Oued Tree Mark AD			
f Romovo LIARI			
f Screen Setting API			
f Set Level State API			
f Set Bain API			
f Set Season API			
f Set Sequence State API			
f Set Snow API			
f Set Time API			
f Set Tool State API			-douber 177
			ILE 12
	送择组件来查看可用事件和函数 这择组件来查看可用事件和函数 1 Kit API 1 Caller 1 Delay Time API 1 Modify Controller Model API 1 Modify Controller API 1 Set Seren Setting API 1 Set Level State API 1 Set Season API 1 Set Season API 1 Set Senow API 1 Set Time API 1 Set Tool State API	遠洋銀件来邀看可用事件和函数	遠洋銀件来邀看可用事件和函数

在全局所有蓝图中均可在右键菜单中找到以下函数
 Execute API:执行单个 API 文件
 Execute API:执行 API 队列
 Execute Tasks:执行自定义任务队列

此蓝图的所有操作	✔ 情境关联 🕨
ex	×
⊿∗ Kit Facade	
⊿Logic	
f Execute UILogic API	line and the second
f Execute API	
$rac{f}{f}$ Execute APIs	
$f_{{ m Ex}}$ ecute Tasks	

- API 功能与参数说明
- 角色控制类
- 切换角色类型和对象 ModifyControllerModelAPI





API 说明: 切换角色类型,并且可以根据 Key 值来指定相应的角色对象,设置 Time 值以改变相机运动速度。

参数名	参数描述	参数类型
Controller Model	填写需要切换的角色类型	鸟瞰角色
		飞行角色
		漫游角色
	填写对应角色上的 Key 值,如为	填写角色 KeylD
Location Name	空则切换到场景中最后生成的对应	
	类型的角色上	
Time	填写切换到该角色时,镜头所运动	填写数值,单位为秒
	的时间,如为0,则角色瞬间切换	

前端调用代码示例

function ModifyControllerModel(ControllerModel, LocationName,Time) {
 let obj = {
 action: "ModifyControllerModel",
 requestId: 10, //开发者自行分配
 params: {
 controllerModel: ControllerModel,
 locationName: LocationName,
 time:Time
 }
 }
 ue4("ExecuteAPIs", obj,TAAPICallbackCommon);

● 切换视角 MoveCameraAPI

f Move Camera API	
D	D
◆ Target Location × 0.0 × 0.0 Z 0.0	
O Target Rotator X 0.0 Y 0.0 Z 0.0	
O ► Target Arm Length 0.0	
O Move Time 0.0	
O Pawn ID	

API 说明: 根据指定的 PawnID 移动角色。

参数名	参数描述	参数类型
Target Location	目标视角位置	填写坐标
Target Rotator	目标视角旋转量	填写旋转度数
Target Arm Length	目标弹簧臂长度(切换到非鸟瞰角 色是忽略此参数)	填写长度值
Move Time	填写切换到该角色时,镜头所运动 的时间,如为0,则角色瞬间切换	填写数值,单位为秒

前端调用代码示例

function MoveCamera(TargetLocation, TargetRotator, TargetArmLenght, MoveTime,PawnId) {
 let obj = {
 action: "MoveCamera",

```
requestId: 10, //开发者自行分配
params: {
    targetLocation: TargetLocation,
    targetRotator: TargetRotator,
    targetArmLenght: TargetArmLenght,
    moveTime: MoveTime,
    pawnId:PawnId
    }
}
```

ue4("ExecuteAPIs", obj,TAAPICallbackCommon);

前端调用代码示例

```
function GetCameraView() {
    let obj = {
        action: "GetCameraView",
        requestId: 10, //开发者自行分配
        params: {
            //无
        }
        ue4("ExecuteAPIs", obj,TAAPICallbackCommon);
```

● 使用视角文件 UseCameraViewAPI

f Use Camera View API	
D	D
Select Asset 👻 🦛 🔎	
🕒 Change Pawn 🔲	

API 说明: 根据指定的视角文件切换视角

参数名	参数描述	参数类型
Camera View	使用视角文件	指定视角文件
Change Pawn	是否切换到创建视角文件时对应的 Pawn,否时将当前角色移动到视 角文件对应的坐标上	是/否

● 界面控制类

● 添加界面 AddUIAPI



API 说明: 根据指定的 UI 类型以及层级根据某种方式添加	U	到视口
---------------------------------	---	-----

参数名	参数描述	参数类型
UIClass	需要添加的 UMG 类(需继承自 Kit Base UMG)	指定 UMG 控件文件
	自定义控件 Show/Hide	自定义类型
Add Type	特殊类型,不接受之后添加的 UI 的隐藏(包括单个显示类型),只 能通过 RemoveUI 移除	TA_菜单类型
	添加后隐藏上一个 UI	TA_隐藏上一个 UI
	添加后叠加在旧的 UI 上面	TA_叠加旧的 UI 上面
	添加后隐藏除该 UI 的所有 UI	TA_单个显示类型
Zorder	UI 层级	填写层级数,数值越大,显示优先 级越高

● 移除界面 RemoveUIAPI



API 说明: 移除当前视口的 UI, 并回到上层

● 跳转界面 ToUIAPI



API 说明:返回到该层级 UI (需要在已创建当前类型 UI 的情况下生效)

参数名	参数描述	参数类型
UIClass	需要添加的 UMG 类(需继承自 Kit Base UMG)	指定 UMG 控件文件

● 设置界面可见性



API 说明:设置指定控件可见性,如隐藏指定控件,需在显示时手动设置可见性,否则将保持可见性为隐藏

参数名	参数描述	参数类型

UMG Class	目标 UMG	填写继承自 kit base umg 的控件
		类
Visibility	可见性	可见类型

● 参数设置类

● 清晰度设置 ScreenSettingAPI



API 说明:设置清晰度(屏幕百分比)

参数名	参数描述	参数类型
Percentage	清晰度百分比,默认值为 100	填写数值

前端调用代码示例 function ScreenSetting(Percentage) { let obj = { action: "ScreenSetting", requestId: 10, //开发者自行分配 params: { percentage: Percentage } } ue4("ExecuteAPIs", obj,TAAPICallbackCommon); }

● 设置时间 SetTimeAPI



API 说明: 基于 SunSky 的时间设置,需要场景中使用 SunSky

参数名	参数描述	参数类型
New Time	目标时间	填写数值(0-24)
During	过渡时间	填写数值,单位为秒

前端调用代码示例

```
function SetTime(NewTime, During) {
    let obj = {
        action: "SetTime",
        requestId: 10, //开发者自行分配
        params: {
            newTime: NewTime,
            during: During
        }
        ue4("ExecuteAPIs", obj,TAAPICallbackCommon);
    }
}
```

音量调节 SoundSettingAPI



API 说明:设置音量		
参数名	参数描述	参数类型
Volume	目标音量,默认值为1	填写数值
		1

	前端调用代码示例
function SoundSetting(Volume) { let obj = { action: "SoundSetting", requestId: 10, //开发者自行分配 params: { volume: Volume } lue4("ExecuteAPIs", obj,TAAPICallba }	ckCommon);

● 延迟 API DelayTimeAPI



API 说明: 延迟, 与其他的 API 配合使用, 可以隔开两个 API 的执行时间或者延迟 执行某个 API

参数名	参数描述		参数类型
Time	延迟时间	填写数值,	单位为秒

● 场景控制类

● 设置关卡状态 SetLevelStateAPI



API 说明:设置关卡状态,设置可见性和加载卸载,加载状态下 Visiable 才生效。

参数名	参数描述	参数类型
Level Names	关卡名称,可填写多个关卡名称, 使用";"(英文分号)隔开	填写关卡名称
Visiable	可见状态,需要先 Load 关卡	可见/不可见

Load	加载状态	加载/卸载

前端调用代码示例	
nction SetLevelState(LevelNames, Visiable, Load) {	
let obj = {	
action: "SetLevelState",	
requestId: 10, //开发者自行分配	
params: {	
levelNames: LevelNames,	
visiable: Visiable,	
load: Load	
}	
ue4("ExecuteAPIs", obj,TAAPICallbackCommon);	

● 设置 Sequence 状态 SetSequenceStateAPI

f Set Sequence State API	
D	D
Sequence Select Asset -	
State 请选择类型 ▼	
O Forward	
O Speed 0.0	
O Target Frame 0.0	
O Loop 0	

API 说明:设置 Sequence 状态。

参数名	参数描述	参数类型
Sequence	指定需要控制的 Sequence 文件	指定 Sequence 文件
State	Sequence 状态操作	播放、暂停、停止、播放到目标帧 数
Forward	播放方向	加载/卸载
Speed	播放速度,默认值为1	填写速度
Target Frame	当 State 选择为播放到目标帧数 时,在此填写目标帧数	填写目标帧数
Loop	循环播放次数,0为无限循环	填写次数

● 控制标签显隐 ShowMarkAPI



API 说明:设置场景中标签的显示隐藏

参数名	参数描述	参数类型
Mark Name	填写 Actor 下 Tag Name 以控制所 有拥有相同 Name 的 Actor,可填 写多个 Name,使用";"(英文分 号)隔开	填写 Tag Name
State	标签显示状态	显示/隐藏

前端调用代码示例

function ShowMark(MarkName, State) {

let obj = { action: "ShowMark",

userId: sessionStorage.getItem("userId"),

```
params: {
	markName: MarkName,
	state: State
	}
}
ue4("ExecuteAPIs",obj,FunctionCallbackCommon);
ue4("ExecuteAPIs", obj,TAAPICallbackCommon);
```

● 设置模型(Static Mesh)组件资源

f	Set Component Mesh API	
D		D
o	םוטט	
0	Mesh 选择资产 🗸 🕞 🔍	
0	Mesh 选择资产 🗸 🕞 🔾	

API 说明: 通过指定 UUID 切换该模型组件引用的资产

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、 widget 组件的 ID	填写 Static Mesh 组件的 ID
Mesh	指定切换的模型资产	模型资产

● 设置模型(Static Mesh)组件变换



API 说明:通过指定 UUID 变换该组件的位置、旋转、缩放,可控制过渡时间、相对 坐标、世界坐标

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、 widget 组件的 ID	填写 Static Mesh 组件的 ID
During	过渡时间	填写数值,单位为秒
Absolute	变换坐标系	绝对坐标/相对坐标
New Transform	目标变换参数值	变换的目标参数

前端调用代码示例		
function SetComponentTransfromById(UUID,During,Absolute,NewTransform) { let obj = {		

```
action: "SetComponentTransfromById",
requestId: 10, //开发者自行分配
params: {
UUID: UUID,
during:During,
absolute:Absolute,
newTransform:NewTransform,
}
ue4("ExecuteAPIs", obj,TAAPICallbackCommon);
```

▶ 设置模型(Static Mesh)组件材质



API 说明: 通过指定 UUID 变换该组件的材质

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、	填写 Static Mesh 组件的 ID
	widget 组件的 ID	
Mat	指定切换材质资产	指定材质资产
Mat Index	指定材质 ID	设置 ID 为-1 时,将为所有材质 ID
		切换指定材质

● 设置模型(Static Mesh)组件可见性



API 说明:通过指定 UUID 切换该组件的可见性

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、 widget 组件的 ID	填写 Static Mesh 组件的 ID
Visable	设置可见性	可见/不可见

前端调用代码示例

```
function SetComponentVisible(UUID,Visible) {
    let obj = {
        action: "SetComponentVisible",
        requestId: 10, //开发者自行分配
        params: {
            UUID: UUID,
            visible:Visible
        }
        lue4("ExecuteAPIs", obj,TAAPICallbackCommon);
```

● 设置模型(Static Mesh)组件材质标量参数





API 说明:通过指定 UUID 切换该组件指定材质 ID 的材质标量参数

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、	填写 Static Mesh 组件的 ID
	widget 组件的 ID	
Param Name	指定参数名称	填写材质中参数化标量的参数名
Param	标量参数目标值	填写材质中参数化标量的目标参数
Mat Index	目标材质的材质 ID	填写目标材质在模型组件中的材质 ID
During	参数变化的过渡时间	填写数值,单位为秒

前端调用代码示例

function SetComponentMaterialScalar(UUID,ParamName,Param,MatIndex,During) {

```
let obj = {
    action: "SetComponentMaterialScalar",
    requestId: 10, //开发者自行分配
    params: {
        UUID: UUID,
        paramName:ParamName,
        param:Param,
        matIndex:MatIndex,
        during:During
    }
}
ue4("ExecuteAPIs", obj,TAAPICallbackCommon);
```

● 设置模型(Static Mesh)组件材质向量参数

🥑 Set Component Material Co	olor API
D	D
ouu 🗿	
O Param Name None	
📀 Param 📃	
🔿 Mat Index 🕕	
O During 0.0	

API 说明:通过指定 UUID 切换该组件指定材质 ID 的材质向量参数

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、 widget 组件的 ID	填写 Static Mesh 组件的 ID
Param Name	指定参数名称	填写材质中参数化向量的参数名
Param	标量参数目标值	填写材质中参数化向量的目标参数
Mat Index	目标材质的材质 ID	填写目标材质在模型组件中的材质 ID

During	参数变化的过渡时间	填写数值,单位为秒	
	1		
	前端调用代码示例	J	
function SetComponentMaterialColor(UUID,ParamName,Param,MatIndex,During) { let obj = { action: "SetComponentMaterialColor", requestId: 10, //开发者自行分配 params: { UUID: UUID, paramName:ParamName, param:Param, matIndex:MatIndex, during:During }			
ue4("ExecuteAPIs", obj, IAAPICallbackCommon);			

● 设置模型(Static Mesh)组件外描边

API
D

API 说明: 通过指定 UUID 开启/关闭模型组件的外描边,可设置自定义深度通道

参数名	参数描述	参数类型
UUID	场景中 Actor、static mesh 组件、 widget 组件的 ID	填写 Static Mesh 组件的 ID
State	开启/关闭外描边	开启/关闭
Custom Depth	指定自定义深度通道(需在项目设 置中开启自定义模板)	当通道值为-1时,则采用模型组件 上设置的自定义深度通道值

● 关闭所有模型(Static Mesh)组件外描边



API 说明:关闭所有模型组件外描边

● 工具控制类

● 切换集成工具 SetToolStateAPI

D		D
0	Tool Type	
	请选择类型 天际线分析工具 通域分析工具 推注工具 推选搜索工具 在选搜索工具 长度测量工具 FBX导入工具(未解锁) 关闭所有集成工具	

参数名	参数描述	参数类型
ТооІТуре	选择需要开启的工具类型	选择需要开的工具

	前端调用代码示例
function SetToolState(ToolType) { let obj = { action: "SetToolState", requestId: 10, //开发者自行分配 params: { toolType: ToolType } ue4("ExecuteAPIs", obj,TAAPICallba }	ickCommon);

密集撒点聚合 API QuadTreeMarkAPI



API 说明:设置撒点密集聚合标签的开关

参数名	参数描述	参数类型
Show Debug	是否绘制调试网格	是/否
Quad Counter Type	指定聚合标签的样式类型,默认会 挂载一个 Widget 显示数量,界面 和模型样式可自定义拓展,详见文 档-插件预设类	指定 Counter 类
Quad Mark Type	指定聚合标签的撒点类,用于定位 标签位置,可自行定义聚合标签生 成方式,详见文档-插件预设类	指定 Mark 类
Max Depth	深度,细分程度,越大分越细	填写深度数值
Draw Alpha	每层的划分步长,越大越密集	填写步长数值
State	当前指定聚合标签类显示状态	显示/隐藏

● 隐藏所有密集撒点聚合 API Hide All QuadTreeMarkAPI



- 如何在管理器中调用自定义 API、函数
- 自定义任务队列(Task)
- 在管理器中,预置 API 并不足以完成所有复杂逻辑,故引入自定义任务队列,自 定义队列可快速在管理器、UMG、Actor 中使用。
- 创建"KitBaseTask"

U	Pick Parent Class	×
▼ 通用		
🧕 Actor	Actor是一种可在世界中放置或动态生成的对象。	0
🛓 Pawn	Pawn是一种可以被"控制"的Actor,且可以接收来自Controller 的输入。	0
👲 角色	角色是Pawn的一种类型,增加了可四处走动的功能。	0
🎮 玩家控制器	PlayerController是Actor的一种子类型,其负责控制玩家所使 用的Pawn。	0
🕮 游戏模式基础	GameModeBase定义了正在进行的游戏、其规则、得分以及 游戏类型的其他方面。	
C Actor组件	ActorComponent是一种可复用组件,能被附加到任意Actor 上。	0
▲。场景组件	SceneComponent是一种组件,能够进行场景变换并可被附加 到其他场景组件下。	0



● TaskActor 中由"Task Finished"、"Execute Task"事件组成,在 Execute Task 事件后,可自定义逻辑。

(1) 文件 編編 资产 重看 调整		- a x
Scene		x : <u>Kit Base Task</u>
■ K0 10 441年 : • 5 対比 >		
El tert ×	:# 祝口 f Construction Scr *: 事件因表 × 2 指称 ×	
+ 添加 Q 注意	■ < ← → NewCharlesBlueprint > 事件图表 缩放1:1	
● NewcharlesBlogrint(自我) Ag DefaultSceneRoot	Construction where the second of the second	
▲ 我的蓝图 ×		
+1010 Q H R 🗘	◆ ₩ #Task Finished □ ◆ 父类: Task Finished	
→田衣		
 ▼ 記事件图表 ◆ 事件开始运行 ◆ 事件不均加量量 ◆ 事件Tick ◆ 事件Execute Task ◆ 事件Task Frished 		
▽函数(19可混差) ④		
⑦ 构造脚本		
# ⊙	(双周开攻	
>交量		
DefaultSceneRoot -	花图	
事件分发器 ③		
	B within a Q states	
■影 内容倒滑菜单 📓 输出日志 💽 Cmd 🔻	1 0ARM:000	R存 🖉 源码管理」

● 在 API 生成工具中或在管理器中创建"执行任务队列"API。

MarkEditorMode	× 🗣 放置Actor
选择对应的编辑模式	式进行使用,场景中点击鼠标左 P!生成工具 ~
🗢 * TA API	
Create New API	
APIName	
APICIass	请选择类型 🗸
当前API实例	提案
L	雨(木爽現) 雪(未实現) 季节(未实現) 时间系统天气系统开关(未实现)

在 API 参数中添加 Task List, 点击加号并填入以上创建的"KitBaseTask"。

A	NPI生成工具 🗸
🐨 * TA API	
Create New API	
APIName	task 🕤
APIClass	执行任务队列 🗸 🕤
当前API实例	
🗢 * TA API	
Task List	0 数组元素 🕤 🔂
🐨 Kit Base API	
APIType	执行任务队列 🗸
🗢 Execute Json Msg	
API_Json	

- 此处 Task List 可填入多个 KitBaseTask 或同一个 KitBaseTask 填入多次, Task 事件将按索引次序执行。
- API 文件可快速在管理器、UMG、场景 Actor 中使用。



- 创建"Actor",创建自定义函数。

文件 編組 资产 查看 词道 业 Scene	BID IJA MBB) NewCharlesBlueprint Project/JLogicTool	- の × 父母: <u>Actor</u>
🗎 📷 🧭編译 : •5 対比 🗸	🗩 直线 🐾 隐藏不相关 : 🛱 类设置 🗶 类默认值 🍉 模拟 🍃 🕨 🔳 📥 : 未这中国运行象 🗸	
EH 细件 ×	## 視口 チ Construction Scr ** 単作田根 チ Customfuntion ×	2 续节 ×
+ ###0 Q 11.11	順 → 🔶 🗲 NewCharlesBlueprint1 > Customfuntion 缩放1:	1
(B) HenCharlesBlurgrint (B)() (B)() (b) DefaultSceneRoot		
局 我的蓝图 ×		
4 mm Q 112		
	Customruntion	
 ▼ 計事件因表 ◆ 事件开始运行 ◆ 事件Actor开始重叠 ◆ 事件Tick 	10228 • • • Free todes [before • • barry 10227 • • • Last index • Completed • • barry Last index • Completed •	
う 构造群本 プ customfunction		
	~ — — — — — — — — — — — — — — — — — — —	
	بحا سا	
	Y WiNBUER × Q EREER ・ (2554.M5/NexOurlesElumprint) 編品(2):(2) 東谷内)(/Same/Elumprint)/NexOurlesElumprint). 	
	2月11日11日11日11日11日11日11日11日11日11日11日11日11日	
		B (set A stress

● 在 API 生成工具中或在管理器中创建"指定 UUID 调用 Actor 自定义方法"API。

MarkEditorMode	× •••	放置Actor		
选择对应的编辑机	莫式进行使月	月,场景中	P点击鼠标	左
	API生成工具	~		
🔻 * TA API				
Create New AP	1			
APIName	custom	API		\$
APIClass	指定UUI	D调用Actor	自定〉 🗸	6
all de la martinet		无	~	
当前API实例	None	e	5	

● 将自定义 Actor 拖入场景中并复制其 UUID 填入 API 参数中, "Function Name"为所需调用的函数,"Args"中添加需传入的参数,其索引对应自定义函数中 的参数索引。

送择对应的编辑模式进行使用,场景中点击줾标左 API生成工具 ~ Create New API APIName CustomAPI APIClass APIClass APIClass APIClass APIClass APIClass APIClass APICUUD询用Actor自定) ~ 5 CustomAPI1 ~ 5 CustomIuntion ~
★ TA API Create New API APIName CustomAPI APIClass 指定UUID调用Actor自定〉 当前API实例 ごいtomAPI1 当前API实例 ごいtomAPI1 ○< 「uUID NewCharlesBlueprint1_C_1 Function Name Customfuntion 「Args 2数组元素 ① 「方 ※ Args 「方 「方
Create New API S APIName CustomAPI S APIClass 指定UUID调用Actor自定) S 当前API实例 CustomAPI1 S 当前API实例 CustomAPI1 S マ・TA API Customfuntion S Function Name Customfuntion S 索引[0] 5 、
APIName customAPI 。 APIClass 指定UUID调用Actor自定) 、 当前API实例 CustomAPI1 、 で、TA API UUID NewCharlesBlueprint1_C_1 、 Function Name customfuntion … Args 2数组元素 ① 1 5
APIClass 指定UUID调用Actor自定) 、 ら 当前API实例 CustomAPI1 、 で、TA API UUID NewCharlesBlueprint1_C_1 、 Function Name Customfuntion
当前API实例 当前API实例 ・TA API UUID NewCharlesBlueprint1_C_1 ・ Function Name customfuntion ・ Args 2数组元素 ① 1 ・ 家引 [0] 5 ・
▼ ★ TA API UUID NewCharlesBlueprint1_C_1 Function Name customfuntion Args 2数組元素 ④ む 家引[0] 5 ✓ 5
UUID NewCharlesBlueprint1_C_1 5 Function Name customfuntion 5 Args 2数组元素 ① む 5 家引[0] 5 く 5
Function Name customfuntion ら Args 2数組元素 ① む ら 索引[0] 5 く
▲ rgs 2 数组元素 ① 立 索引[0] 5 ✓
索引[0] 5 🗸 🔶
索引[1] 哈哈哈哈哈哈 🗸 😽
🗢 Kit Base API
APIType 指定UUID调用Actor自定》
🔻 Execute Json Msg
API_Json

● API 文件可快速在管理器、UMG、场景 Actor 中使用。

TechnologyArt-Implementer

● 基本概述

每个 API 带有一个实现器,方便二次开发使用,比如切换角色 API 的实现器



● 使用方式

叠加实现

可以利用,绑定自定义事件来增加 API 的实现,会在原有基础上进行叠加:

	f Modify Controller Mode	el Implementer
	D	C
	Callback	
CustomEvent_1		
D		
Controller Model		



重写(覆盖)实现

如果需要重写原先的 API 实现 可以全局搜索该实现器,即可跳转到原先的实现



在蓝图列表中可以轻松获取到 API 的调用器和实现器,他们是一一对应的:



All Actions for this Blueprint	🌌 Context Sensitive 🕨
Search	Q
Select a Component to see available Ev	ents & Functions
▲• Kit API	
⊿ Caller	
f Add UIAPI	
$oldsymbol{f}$ Delay Time API	
f Modify Controller Model API	
f Move Camera API	
f Quad Tree Mark API	
f Remove UIAPI	
f Screen Setting API	
f Set Level State API	
f Set Rain API	
f Set Season API	
\overline{f} Set Sequence State API	
f Set Snow API	
f Set Time API	
f Set Tool State API	
Col er Line	

实现器:

All Actions for this Blueprint	🖌 Context Sensitive 🕨
Search	Q
Select a Component to see available Ev	ents & Functions
▲• Kit API	
D Caller	
▲ Implementer	
f Add UI Implementer	
$oldsymbol{f}$ Delay Time Implementer	
f Modify Controller Model Impleme	enter
f Move Camera Implementer	
$oldsymbol{f}$ Quad Tree Mark Implementer	
$oldsymbol{f}$ Remove UI Implementer	
$oldsymbol{f}$ Screen Setting Implementer	
$oldsymbol{f}$ Set Level State Implementer	
$oldsymbol{f}$ Set Rain Implementer	
$oldsymbol{f}$ Set Season Implementer	
f Set Sequence State Implementer	
f Set Snow Implementer	
f Set Time Implementer	
🖉 Oat Taal Otata Imalamantar	

TechnologyArt-WebUI 拓展

- 下载页 <u>https://www.tartcloud.cn/download</u>
- 支持插件版本 V1.0.6 以上的版本

V1.0.61	- 插件WebUI拓展	
---------	-------------	--

5.0.X

土

- 解压完成后有以下内容
- 将 Marketplace 拷贝到 UE 引擎根目录的对应文件夹下即可按照成功

用户 → Charlesvane → 文档 → Downloads → WebUIForTA

名称 ^	修改日期	类型	大小
cef_binary_84.1.6+gc551bc2+chromi	2022/6/19 18:16	文件夹	
Marketplace	2022/8/12 0:31	文件夹	
Win64	2022/2/23 10:55	文件夹	

● 打开引擎打开对应的插件



● 拓展内容在插件文件夹下

▼ DevKitsDemo	Q
💌 🎥 All	
🔋 🕨 💼 Content	
 Image: C++ Classes Plugins Image: Son Library C++ Classes Image: TECHNOLOGY ART Content Image: TECHNOLOGY ART C++ Classes 	

● 将里面的 WebUIManager 挂在到 Controller 上





webCode 参考以下附件: <u>sample.html</u>

以下用一个示例方法来演示 只是设置

\$("#ShowMark").click(function(e) { console.log("call API"); var APIStruct=new Object(); APIStruct.Action="ShowMark"; APIStruct.RequestId=10; var APIParams=new Object(); APIParams.MarkName="common"; APIParams.state=true;

```
APIStruct.Params=APIParams;
// executed in blueprints
ue4("ExecuteAPIs", APIStruct,TAAPICallbackCommon);
```

```
});
```

//通用接口:

```
//标签状态设置
/*
markName (string) : 标签名称
state (bool) : 是否显示
*/
function ShowMark(MarkName, State) {
    let obj = {
        action: "ShowMark",
        userId: sessionStorage.getItem("userId"),
        params: {
            markName: MarkName,
            state: State,
        }
    }
    ue4("ExecuteAPIs",obj,FunctionCallbackCommon);
    console.log(obj);
}
```

只要通过对应的 Action 和 params 就可以调用 api: 相应的 UEAPI 如下:



只要在 UIConnect 上面添加新的方法就可以用一样的方式使用 js 来调用:



TechnologyArt-CEF 内核替换

上个章节下载的文件夹内部有另外两个文件夹就是 CEF 内核的编译产物 只有更新了 内核才能播放视频和使用高版本的前端组件。

称 ^	修改日期	类型		大小	
cef_binary_84.1.6+gc551bc2+chromi	2022/6/19 18:16	文件夹			
Marketplace	2022/8/12 0:31	文件夹			
Win64	2022/2/23 10:55	文件夹			
别替换以下文件夹即可: am Files 》 Epic Games 》 UE 5.0 》 End	nine > Binaries > T	hirdParty → CF	F3 >		
	gine y binanes y n				
A称	修改日期	类型		大小	
名称 ^	修改日期 2022/2/23 10:56	类型 文件夹		大小	
名称 Linux Win64	修改日期 2022/2/23 10:56 2022/7/17 13:40	类型 文件夹 文件夹		大小	
名称 Linux Win64 ChromiumEmbeddedFramework.tps	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29	类型 文件夹 文件夹 TPS 文件		大小	КВ
名称 Linux Win64 ChromiumEmbeddedFramework.tps	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29	类型 文件夹 文件夹 TPS 文件		大小	КВ
名称 Linux Win64 ChromiumEmbeddedFramework.tps aux ma Work (Ei) > Program Files > Epic Games > UE_5.0 > Engine	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29	类型 文件夹 文件夹 TPS 文件		大小	КВ
各称 Linux Win64 ChromiumEmbeddedFramework.tps 题,Work (E:) > Program Files > Epic Games > UE_5.0 > Engine 名称	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29 f)// usre e > Source > ThirdParty > CEF3	类型 文件夹 文件夹 TPS 文件	类型	大小 1 _{大小}	КВ
各称 Linux Win64 ChromiumEmbeddedFramework.tps 题,Work (Ei) > Program Files > Epic Games > UE_5.0 > Engine 名称 ^ 	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29 7)// 2029 7)// 2029 40.414738.windows64	类型 文件夹 文件夹 TPS 文件 ^修 农日期 2022/6/19 18:16	美型 文件夹	大小 1 ^{大小}	КВ
各称 Linux Win64 ChromiumEmbeddedFramework.tps 题 Work (E) > Program Files > Epic Games > UE_5.0 > Engine 名称 cef_binary.84.1.6+gc551bc2+chromium-84 cef_binary.94.4.1+gracr.504+chromium-84	修改日期 2022/2/23 10:56 2022/7/17 13:40 2022/2/23 10:29 7)// 2029 7)// 2029 700 700 700 700 700 700 700 700 700 70	类型 文件夹 文件夹 TPS 文件 ^修 改日期 2022/6/19 18:16 2022/27 10:03	英型 文件夹 文件夹 文件夹 文件	大小 1 ^{太小}	KB