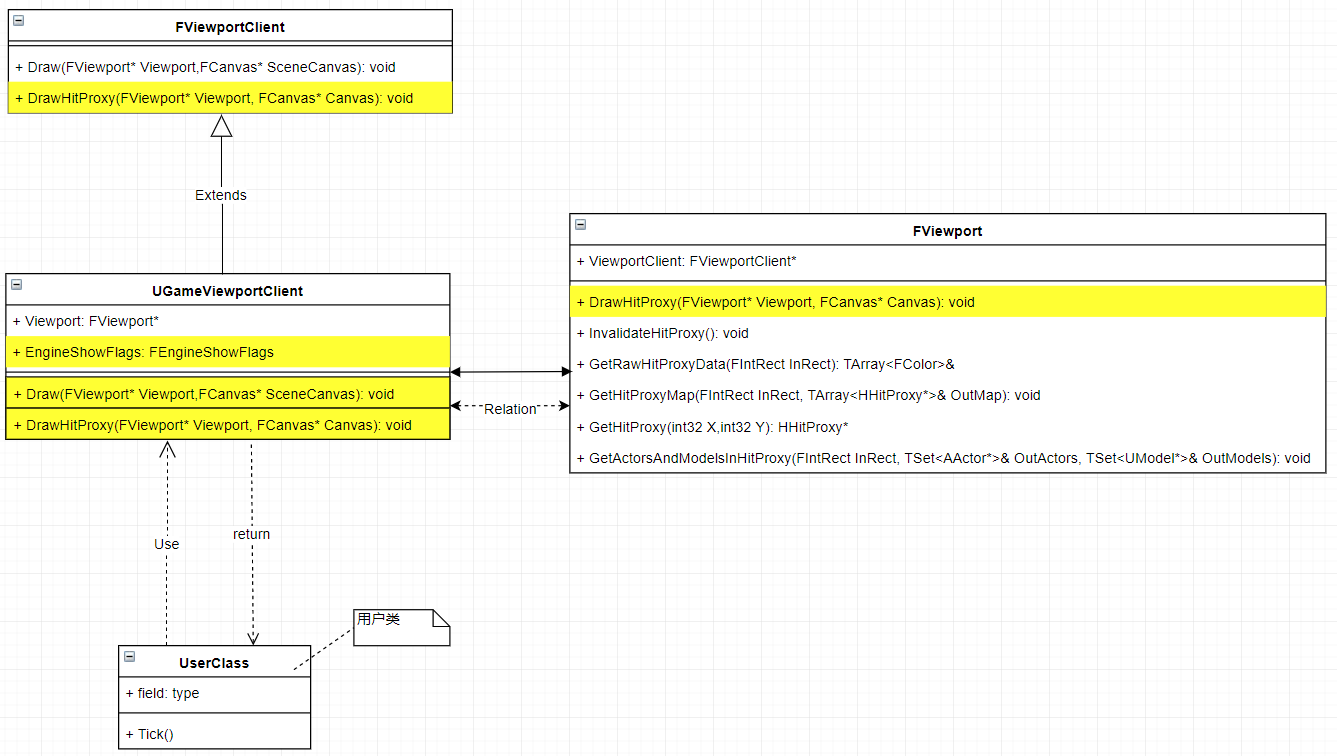
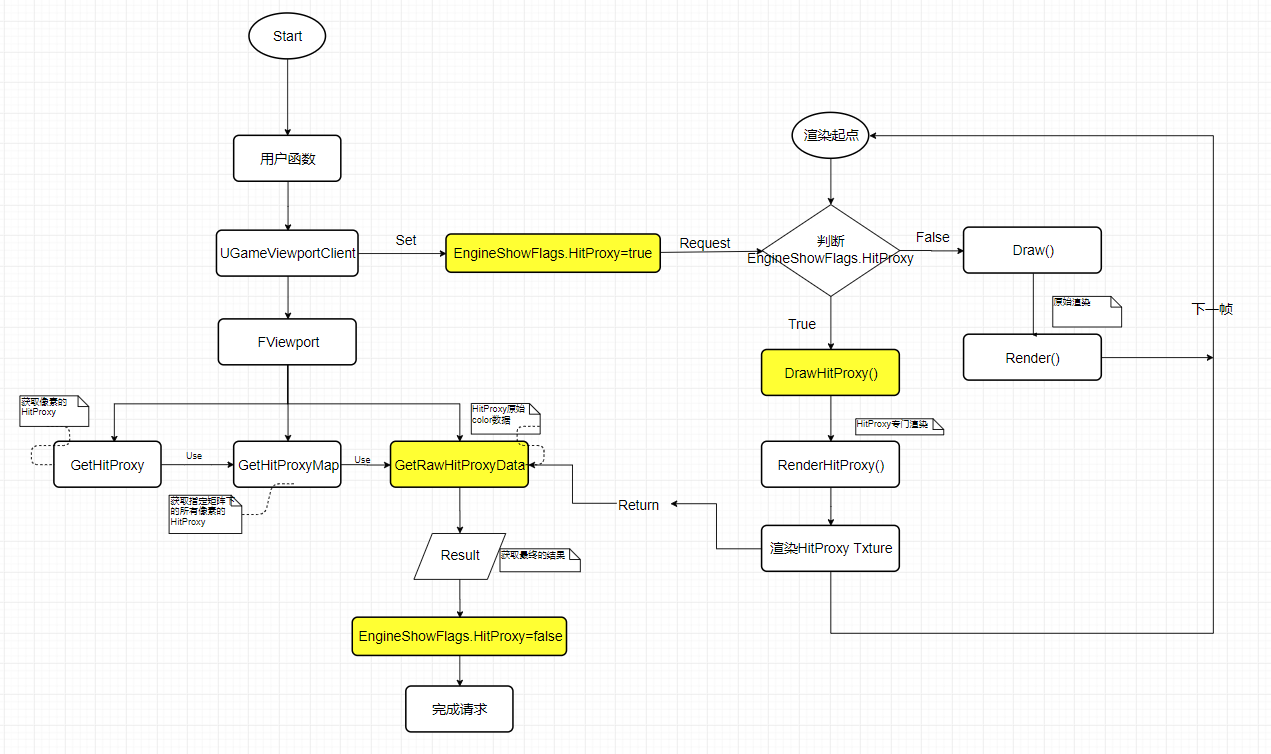
# 使用HitProxy

类图：



HitProxy\_UML

流程图：



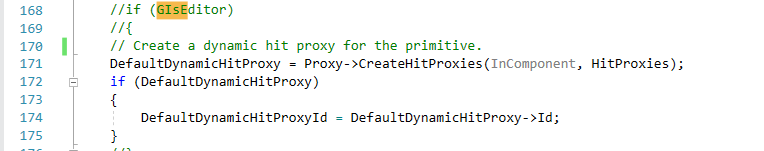
流程图

## Game模式使用HitProxy

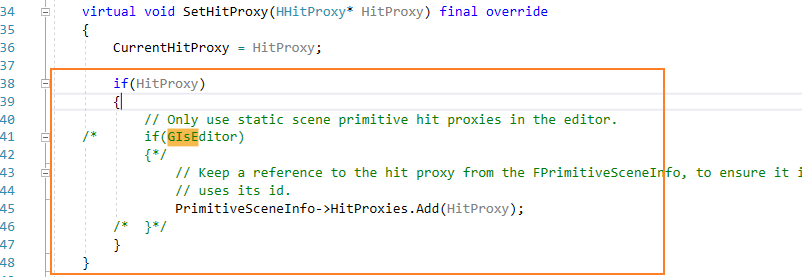
### 删除相关WITH\_EDITOR宏

#### 文件Engine\Source\Runtime\Renderer\Private\PrimitiveSceneInfo.cpp

在创建初始化HitProxy的下面的函数注释掉WITH\_EDITOR宏

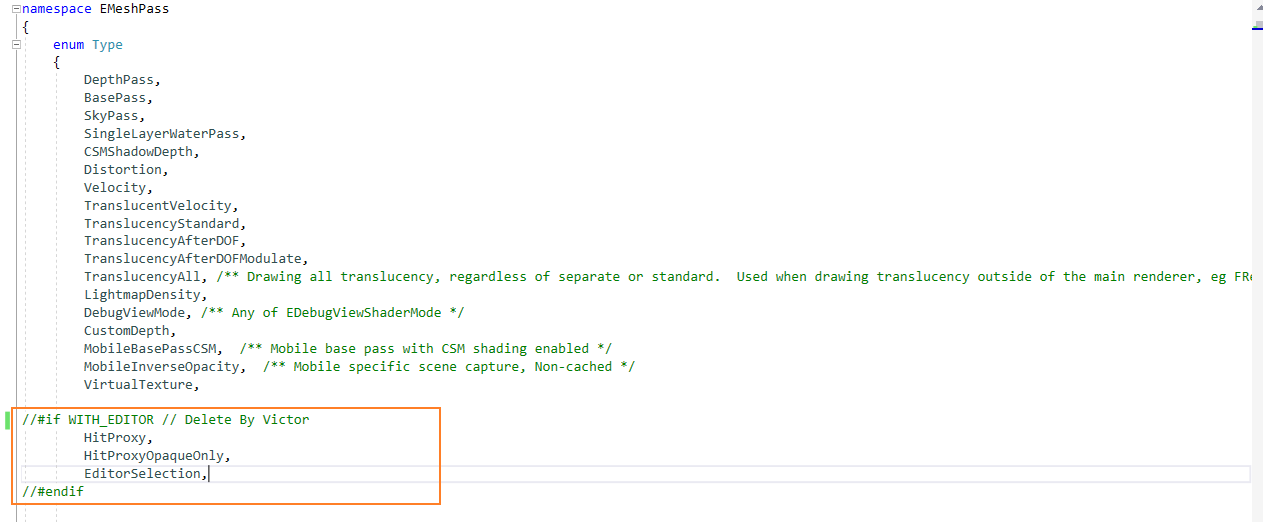


SetHitProxy函数注释掉WITH\_EDITOR宏



#### 文件Engine\Source\Runtime\Renderer\Public\MeshPassProcessor.h

声明MeshPass的地方注释掉WITH\_EDITOR宏

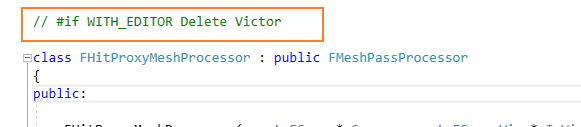


在对应的名字匹配的地方也注释掉WITH\_EDITOR宏



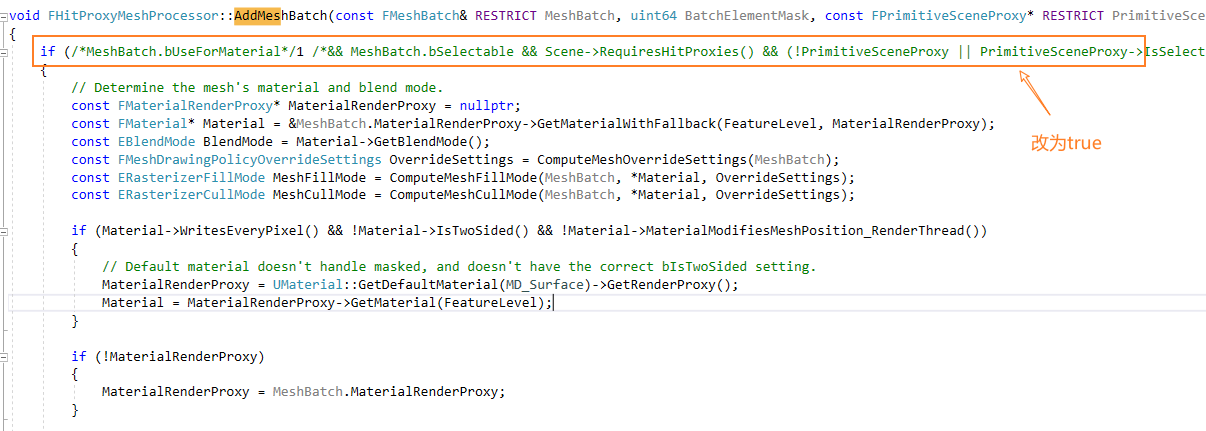
#### 文件Engine\Source\Runtime\Renderer\Private\SceneHitProxyRendering.h

去除类FHitProxyMeshProcessor的WITH\_EDITOR宏约束，去除该文件下的所有WITH\_EDITOR宏



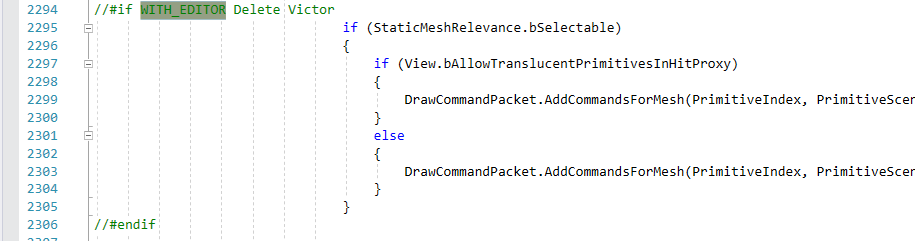
#### 文件Engine\Source\Runtime\Renderer\Private\SceneHitProxyRendering.cpp

去除该文件下的所有WITH\_EDITOR宏，并且修改下面的判断：

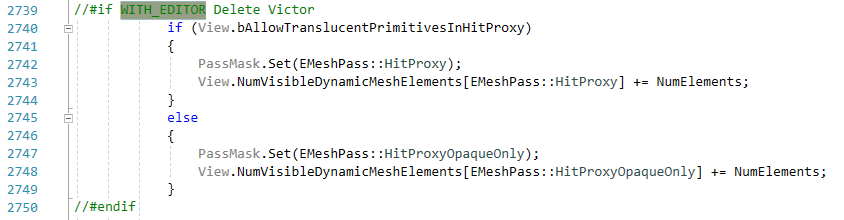


#### 文件Engine\Source\Runtime\Renderer\Private\SceneVisibility.cpp

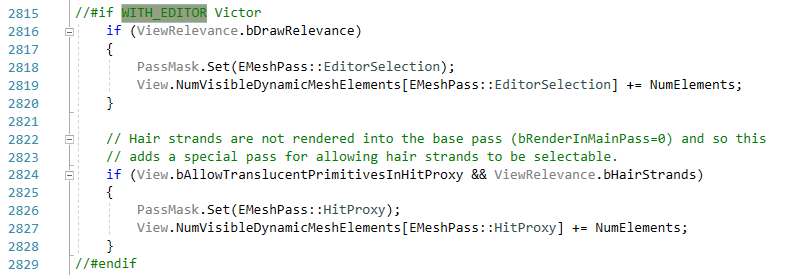
在HitProxy Pass添加静态网格的地方去掉WITH\_EDITOR宏



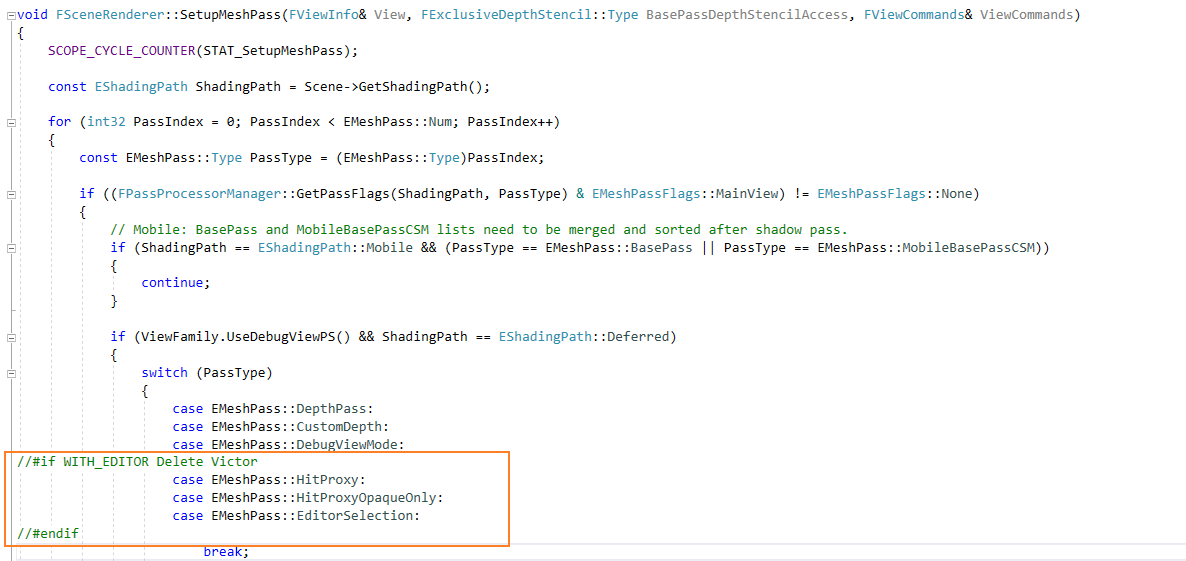
在HitProxy Pass添加动态网格的地方去掉WITH\_EDITOR宏



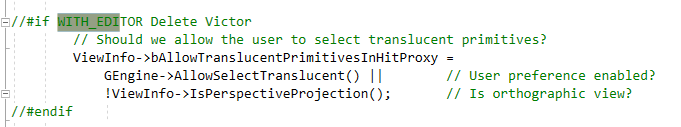
其他需要取消注释的地方



#### 文件Engine\Source\Runtime\Renderer\Private\SceneRendering.cpp



其他需要取消注释的地方



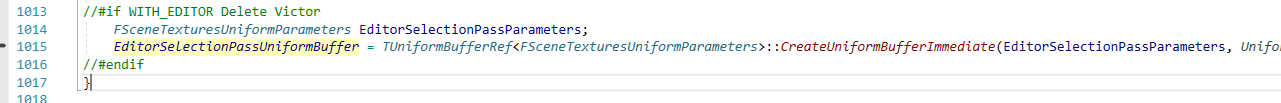
#### 文件 \Engine\Source\Runtime\Renderer\Private\ScenePrivate.h

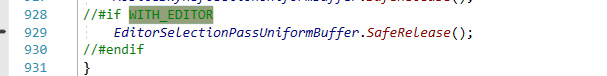
去除WITH\_EDITOR

image-20200705102705329

#### 文件Engine\Source\Runtime\Renderer\Private\RendererScene.cpp

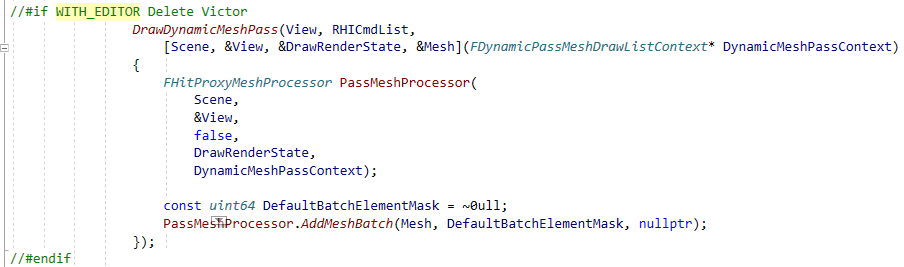
去掉注释





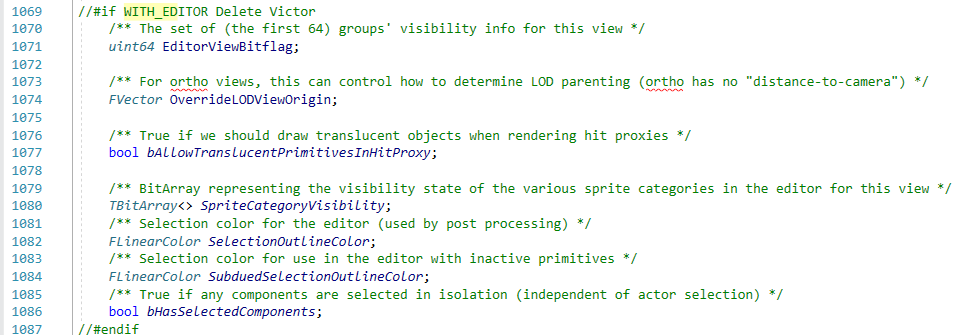
#### 

#### 文件Engine\Source\Runtime\Renderer\Private\Renderer.cpp



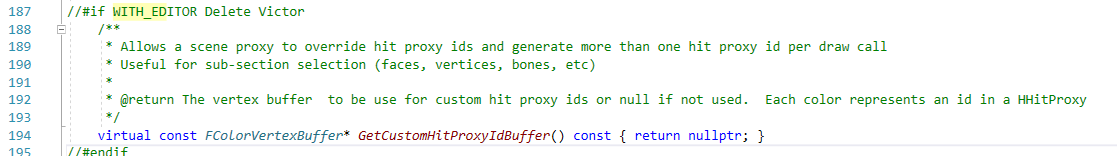
#### 

#### 文件Engine\Source\Runtime\Engine\Public\SceneView.h



#### 

#### 文件Engine\Source\Runtime\Engine\Public\PrimitiveSceneProxy.h



#### 

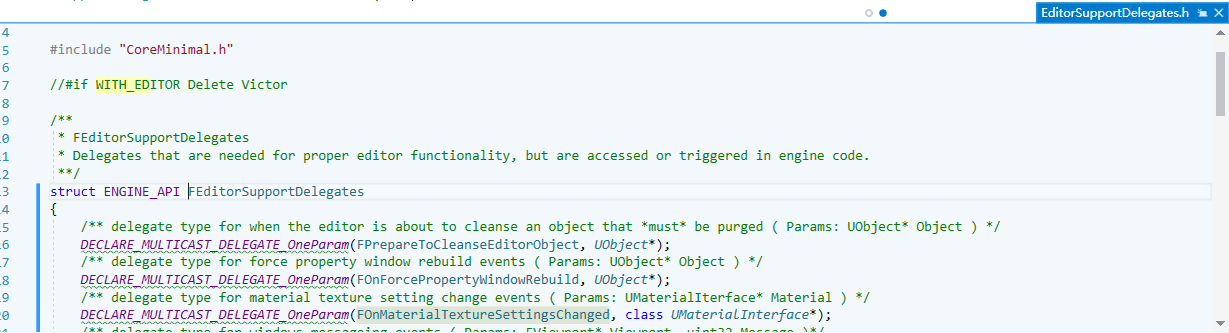
#### 文件Engine\Source\Runtime\Engine\Private\EngineGlobals.cpp

取消注释

image-20200705103914680

#### 

#### 文件：Engine\Source\Runtime\Engine\Public\EditorSupportDelegates.h的宏



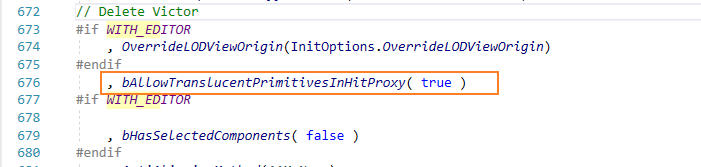
#### 

#### 文件Engine\Source\Runtime\Engine\Private\UnrealClient.cpp



#### 

#### 文件Engine\Source\Runtime\Engine\Private\SceneView.cpp



### 修改源码部分

#### 在FViewportClient类中新建DrawHitProxy函数

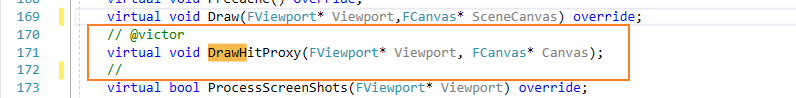
文件UnrealClient.h



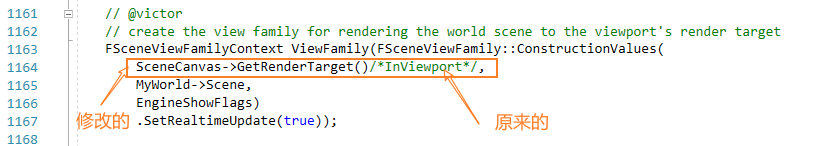
#### 

#### 在GameViewportClient类中声明并且实现

声明：\Source\Runtime\Engine\Private\GameViewportClient.h



将GameViewportClient类中的函数Draw()内容复制到该函数DrawHitProxy，修改下面的的地方：



#### 

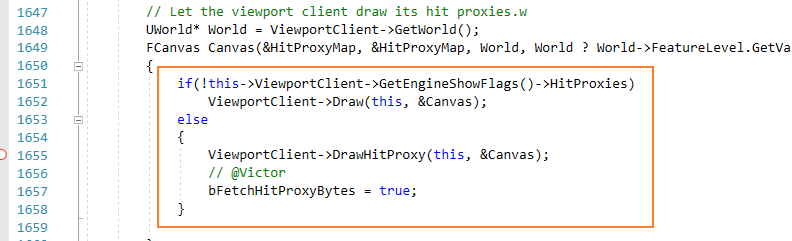
#### 在GameViewportClient类中重写下面的函数

修改返回值为true,路径：\Source\Runtime\Engine\Private\GameViewportClient.h

virtual bool RequiresHitProxyStorage() override { return true; }

#### 修改FViewport类中的GetRawHitProxyData函数

在GetRawHitProxyData函数中进行以下的修改：Engine\Source\Runtime\Engine\Private\UnrealClient.cpp



### 调用

可以使用以下的代码

UGameViewportClient\* gameViewport = GWorld->GetGameViewport();

TArray<FColor> outData;  
FIntRect rect(0, 0, viewportSize.X, viewportSize.Y);  
TSet<AActor\*> OutActors;  
TSet<UModel\*> OutModels;  
// 设置获取HitProxy的请求  
GWorld->GetGameViewport()->GetEngineShowFlags()->HitProxies = true;  
gameViewport->Viewport->InvalidateHitProxy(); // 请求刷新HitProxy  
  
// 获取指定像素的HitProxy  
HHitProxy\* hitProxy = gameViewport->Viewport->GetHitProxy(viewportSize.X / 2, viewportSize.Y / 2);  
// 获取当前视口下所有可见的Actor与Models  
// gameViewport->Viewport->GetActorsAndModelsInHitProxy(rect, OutActors, OutModels);   
// 获取指定矩阵下的所有像素的HitProxy  
// TArray<HHitProxy\*> outMap;  
// gameViewport->Viewport->GetHitProxyMap(FIntRect(100, 100), outMap);   
// 关闭HitProxy的请求  
GWorld->GetGameViewport()->GetEngineShowFlags()->HitProxies = false;  
if(hitProxy)  
 GEngine->AddOnScreenDebugMessage(-1, 5, FColor::Red, FString::Printf(TEXT("HitProxy Name: x: %s;Pos (%f,%f)"), \*static\_cast<HActor\*>(hitProxy)->Actor->GetName(), MousePosition.X, MousePosition.Y));