按照以前，广播应该写在BaseAttributeSet中，但5.3之后有些改变。现在广播写在BaseCharacter中。

首先在BaseCharacter.h中申请一个代理

DECLARE\_DYNAMIC\_MULTICAST\_DELEGATE\_OneParam(FOnHealthChangeEvent,float,NewHealth);

只有多波动态委托才能被蓝图调用。其实就是蓝图里的事件分发器。

其中的FOnHealthChangeEvent这个名字是自己取的，而且必须要以F开头。

然后在类定义中，public中创建对应的变量

UPROPERTY(BlueprintAssignable, Category="Ability")  
FOnHealthChangeEvent HPChangeEvent;

然后进入.cpp文件，实现BeginPlay函数。

void ABaseCharacter::BeginPlay()  
{  
 Super::BeginPlay();  
 UAbilitySystemComponent\* MyAbilitySystemComponent = this->FindComponentByClass<UAbilitySystemComponent>();  
 if(MyAbilitySystemComponent)  
 {  
 MyAbilitySystemComponent->GetGameplayAttributeValueChangeDelegate(UBaseAttributeSet::GetHPAttribute()).AddUObject(this,);  
 }  
}

这里，AddUObject的第二个参数应该是&ABaseCharacter::里的某个事件。其实就是一个函数。

在BaseCharacter.h中，在刚创建的HPChangeEvent变量下面

void OnHealthAttributeChanged(const FOnAttributeChangeData& Data);

实现该函数

我们调用这个函数希望他能干吗？肯定就是进行广播嘛。定义的委托就是这个时候用的。

void ABaseCharacter::OnHealthAttributeChanged(const FOnAttributeChangeData& Data)  
{  
 HPChangeEvent.Broadcast(Data.NewValue);  
}

然后继续实现BeginPlay函数。

MyAbilitySystemComponent->GetGameplayAttributeValueChangeDelegate(UBaseAttributeSet::GetHPAttribute()).AddUObject(this,&ABaseCharacter::OnHealthAttributeChanged);

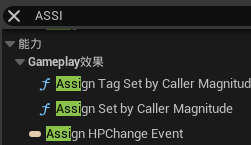
注意，这里OnHealthAttributeChanged，是没有括号的，虽然他是个函数名。

在.h中加上头文件#include "GameplayEffectTypes.h"

在.cpp中加上头文件#include "BaseAttributeSet.h"和#include "AbilitySystemComponent.h"

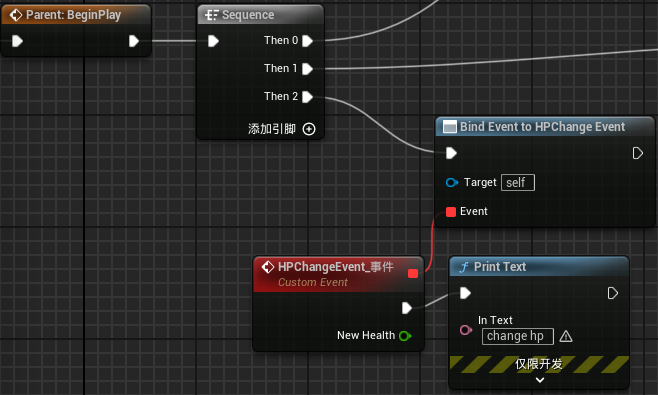
这样，委托就做好了。

回到引擎，来使用这个委托。进入BP\_BaseCharacter



在EventBeginPlay中





此时修改了生命值，就会Print一次。