**Порядок действий для добавления нового элемента в абилки перса.**

1. Создайте класс абилки наследником от UActorComponent в папке

Source/MagicTrigger/AbilitySystem/NewAbility/ . Назовите класс NewAbilityComponent, где NewAbility – название новой абилки.

В этом классе будет описана логика работы абилки.

1. В Файле NewAbilityComponent.h:
   1. Проставьте макрос UCLASS(ClassGroup = (Custom), meta = (BlueprintSpawnableComponent))
   2. Добавьте объявление class AAbilitySystemManager;
   3. Добавьте объявление private AAbilitySystemManager\* AbilitySystemManager;
   4. Добавьте объявление метода void Use();
2. В файле NewAbilityComponent.cpp:
   1. Добавьте инклуды

#include "MagicTrigger/AbilitySystem/AbilitySystemManager.h"

#include "MagicTrigger/Data/DebugMessage.h"

* 1. В метод BeginPlay() добавьте строки:

Super::BeginPlay();

AActor\* Owner = GetOwner();

if (!Owner)

{

DEBUGMESSAGE("!Owner");

return;

}

AbilitySystemManager = Cast<AAbilitySystemManager>(Owner);

if (!AbilitySystemManager)

{

DEBUGMESSAGE("!AbilitySystemManager");

return;

}

* 1. Добавьте пустое определение метода void UNewAbilityComponent::Use()

1. В структуру Source/MagicTrigger/Data/AbilitySystem/AvaliabilityAbilitiesStruct.h добавьте

4.1. UPROPERTY(VisibleAnywhere, BlueprintReadOnly, Category = "AvaliabilityAbilitiesStruct")

bool bNewAbility;

4.2. В конструктор добавьте bNewAbility = true; или bNewAbility = false;

Если true, то на абилку можно переключиться в игре. Иначе – нельзя.

1. В енум Source/MagicTrigger/Data/AbilitySystem/CurrentAbilityEnum.h добавьте

, ECA\_NewAbility UMETA(DisplayName = " NewAbility ")

1. В файл Source/MagicTrigger/AbilitySystem/AbilitySystemManager.h

6.1. Добавьте объявление класса class UNewAbilityComponent;

6.2. Добавьте объявление указателя на объект

UPROPERTY(EditDefaultsOnly, BlueprintReadOnly, Category = "AbilitySystemManager|Abilities")

public UNewAbilityComponent \* NewAbilityComponent;

6.3 Добавьте объявление метода

/\*\*

\* NewAbility

\*/

void UseNewAbility();

1. В файл Source/MagicTrigger/AbilitySystem/AbilitySystemManager.cpp

7.1. Добавьте инклуды

#include "MagicTrigger/AbilitySystem/NewAbility/NewAbilityComponent.h"

#include "MagicTrigger/UI/AbilitySystem/NewAbilityUserWidget.h"

NewAbilityUserWidget будет создан ниже.

7.2. В конструктор класса добавьте создание компонента

NewAbilityComponent = CreateDefaultSubobject<UNewAbilityComponent>(TEXT("NewAbilityComponent"));

7.3. В метод void AAbilitySystemManager::Attack() в switch добавьте

case ECurrentAbility::ECA\_ NewAbility:

NewAbilityComponent->Use();

break;

7.4. В метод void AAbilitySystemManager::SetCurrentAbility(ECurrentAbility InCurrentAbility)

7.4.1. в switch (InCurrentAbility) добавьте

case ECurrentAbility::ECA\_NewAbility:

if (!AvaliabilityAbilities.bNewAbility)

{

return;

}

HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->NewAbilityBorder->SetBrushColor(HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->ActiveBorderColor);

HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->NewAbilityUserWidget->SetColorAndOpacity(HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->ActiveAbilityWidgetColorAndOpacity);

break;

7.4.2. В switch (CurrentAbility) добавьте

case ECurrentAbility::ECA\_NewAbility:

HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->NewAbilityBorder->SetBrushColor(HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->InactiveBorderColor);

HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->NewAbilityUserWidget->SetColorAndOpacity(HUD->PlayerGUIUserWidget->PanelAbilityUserWidget->InactiveAbilityWidgetColorAndOpacity);

break;

7.5. Определите метод void AAbilitySystemManager::UseNewAbility()

например так:

{

PlayerCharacter->AnimationManagerComponent->NewAbilityAnimation();

}

Это значит, что действие абилки будет начинаться с анимации. Пример работы подобной абилки см. в “\MagicTriggerDocumentation\AbilitySystem\SevenfoldSphere\SevenfoldSphereComponent - Логика работы” или других подобных описания.

1. В файле NewAbilityComponent.cpp

определите метод void UNewAbilityComponent::Use()

{

//Какие-то условия, запрещающие запускать абилку.

AbilitySystemManager->UseNewAbility();

}

1. В папке Source/MagicTrigger/UI/AbilitySystem

создайте класс с названием NewAbilityUserWidget наследником от класса

UParentUserWidgetMT

1. В файле Source/MagicTrigger/UI/AbilitySystem/NewAbilityUserWidget.h

10.1. Добавьте объявление класса class UImage;

10.2. Добавьте поле

public:

UPROPERTY(VisibleDefaultsOnly, BlueprintReadWrite, Category = "NewAbilityUserWidget")

UImage\* IconImage;

Это минимум, который необходим, остальные поля по необходимости.

1. В папке

/Content/MagicTrigger/UI/AbilitySystem/

Создайте виджет WBP\_NewAbility.uasset наследником от UNewAbilityUserWidget

Либо, что быстрее, скопируйте другой подобный виджет и перенастройте его родителя. Поменяйте названия в Designer, чтобы они сошлись с названием абилки. В

EventOnInitialized в EventGraph должны быть ноды

Set\_IconImage(IconImage\_C), и другие подобные, чтобы можно было работать с элементами из Designer класса WBP\_NewAbility в классе NewAbilityUserWidget.

1. В файле

Source/MagicTrigger/UI/AbilitySystem/PanelAbilityUserWidget.h

12.1. Добавьте объявление класса class UNewAbilityUserWidget;

12.2. Добавьте объявление элемента

UPROPERTY(VisibleDefaultsOnly, BlueprintReadWrite, Category = "PanelAbilityUserWidget")

UBorder\* NewAbilityBorder;

12.3. Добавьте объявление указателя

UPROPERTY(VisibleDefaultsOnly, BlueprintReadWrite, Category = "PanelAbilityUserWidget")

UNewAbilityUserWidget\* NewAbilityUserWidget;

1. В файле

Source/MagicTrigger/UI/AbilitySystem/PanelAbilityUserWidget.cpp

13.1. Добавьте инклуд

#include "MagicTrigger/UI/AbilitySystem/NewAbilityUserWidget.h"

13.2. В методе void UPanelAbilityUserWidget::SetWidgetsColor()

добавьте настройки

NewAbilityBorder->SetBrushColor(InactiveBorderColor);

NewAbilityUserWidget->SetColorAndOpacity(InactiveAbilityWidgetColorAndOpacity);

либо

NewAbilityBorder->SetBrushColor(ActiveBorderColor);

NewAbilityUserWidget->SetColorAndOpacity(ActiveAbilityWidgetColorAndOpacity);

В зависимости от того, что вы хотите: чтобы в начале игры абилка была активна или не активна. Увяжите этот выбор с выбором в п. 4.2. Если абилка неактивна, то это значит, что ее виджет будет тусклым, и на нее нельзя будет переключиться.

1. В файле

/Content/MagicTrigger/UI/AbilitySystem/WBP\_PanelAbility

14.1. В Designer создайте по типу существующих NewAbility\_Border\_C и добавьте туда виджет WBP\_NewAbility, назовите его WBP\_NewAbility\_C

14.2. В EventGraph в EvenOnInitialized определите переменные NewAbilityBorder и NewAbilityUserWidget по типу существующих определений.

15. В BP\_AbilitySystemManager можно настроить ваш NewAbilityComponent, если в нем вы создавали настройки с макросом

UPROPERTY(EditDefaultsOnly, BlueprintReadOnly, Category = "NewAbilityComponent")

1. Теперь можно скомпилировать все и далее заполнять вашу абилку логикой работы.