**3. UWeaponFXComponent**

1. Что создаем за компонент, от кого наследуемся? Что нужно сделать, чтобы получить доступ к Niagara?

2. Какие функцию и проперти добавили нашему компоненту? В каком ЗФ необходимые функции для спауна? Какую функцию для спауна используем и что она принмает (4 параметра передали мы, один (но не единственный) остался важный по умолчанию – какой)?

3-4. Кому теперь данный компонент и какую логику реализуем (создание, проверки, сама генерация эффекта в какой функции)?

5. У нас наблюдается неправильная ориентация нашего эффекта (частицы генерируются вверх). Как исправить?

1. Первым делом создадим VFX-компонент STUWeaponFXComponent, дополнительный компонент, который будет отвечать за спецэффекты оружия. Добавляем в папку Weapon к нашим оружиям папку Components и в ней создаем, наследуясь от ActorComponent.

Не забываем добавлять новую папку в путь в Build.cs-файле. В этом же файле для использования Niagara надо добавить ее в PublicDependecyModuleNames:

Изображение выглядит как текст

Автоматически созданное описание

2. Нашему новому компоненту добавим функцию и проперти с эффектом, который мы хотим заспавнить (в данном проперти мы будем сетапить ниагара-систему из эдитора):

Изображение выглядит как текст

Автоматически созданное описание

Необходимые функции для спауна находятся в ЗФ NiagaraFunctionLibrary.h:

Изображение выглядит как текст

Автоматически созданное описание

В данной библиотеке есть специальная static-функция, которая умеет спавнить ниагара-систему в мир.

Второй параметр – указатель на ниагара-систему.

Третий – локация в мире.

Четвертый – ориентация в пространстве. Воспользовались полем ImpactNormal параметра Hit, тут хранится нормаль Hit, и создали объект ротатора через Rotation.

Остальные по умолчанию. Один из них – Auto Destroy – стоит true, то есть после того, как ниагара-система закончила анимацию, она автоматически удалится из мира.

3. Создаем новый компонент в классе Rifle.

Изображение выглядит как текст

Автоматически созданное описание

Нам необходим конструктор класса для создания компонента, поэтому его добавляем, так же добавим BeginPlay, в ней сделаем check.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

В функции MakeShot мы убираем рисование линии и сферы и вызываем функцию PlayImpactFX:

Изображение выглядит как текст

Автоматически созданное описание

4. Аналогичным образом добавляем WeaponFXComponent в класс Projectile (так же добавляем компонент, подключаем ЗФ, создаем в конструкторе и проверяем компонент на BeginPlay) – в этом и проявляется архитектурная мощность компонентов – мы можем добавить их как к оружию, так и к логическим частям оружия.

Столкновения с объектом внешней среды у нас происходит в функции OnProjectileHit, поэтому спаун спецэффекта сюда и добавим:

Изображение выглядит как текст

Автоматически созданное описание

Теперь осталось засетапить наши эффекты в Rifle и Projectile (не Launcher!).

5. У нас наблюдается неправильная ориентация нашего эффекта (частицы генерируются вверх). Для исправления этого нужно в эмиттере поменять в параметре Add Velocity in Cone ориентацию с XYZ: 001 на XYZ: 100. Теперь они генерируются вдоль нормали, которая приходит в HitResult.