**4. Impacts, физические материалы**

1. Какие создали импакты из ранее созданного NS\_BaseImpact и как их настроили?

2. Создали новую папку и добавили в них физические материалы – для чего и что это такое и какие вообще создали? Как настроили материалы нашему персонажу?

3. Что нужно подключить и куда, чтобы получить доступ к нашим физическим материалам? Что создали для сопоставления материалов и импактов? Какую логику реализовали в проигрывании импакта?

4. Какие настройки и кому нам надо добавить, чтобы получать наши материалы?

5. Что теперь надо настроить в наших BP-классах?

6. Все-таки Projectile – это особый случай, сделаем для него одинаковый эффект для всех поверхностей. Как и какой?

1. Будем спаунить различные импакт-эффекты в зависимости от того, куда мы попали.

Переименовали ранее созданный NS\_BaseImpact в NS\_DefaultImpact, и от этого создали еще Ground, Body и Head и выбрали для всех различные цвета (а для хедшота еще и большое количество частиц).

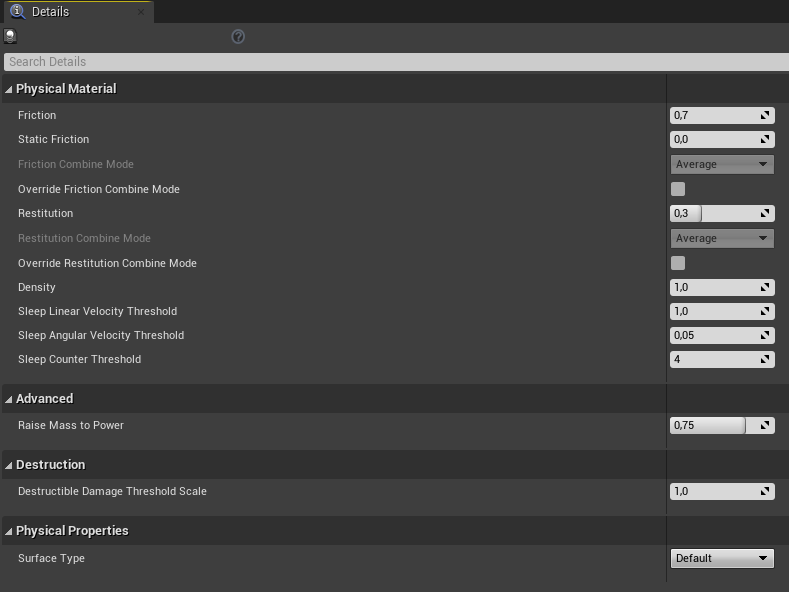
Изображение выглядит как текст

Автоматически созданное описание

2. Создадим новую папку PhysMaterials – рассмотрим новый тип ассета Physical Material – данный вид ассета используется для задания физических характеристик объекта. Чтобы его создать, нажимаем ПКМ-Physics-PhysicalMaterial. Создадим три материала – PhysMat\_Ground, PhysMat\_Head, PhysMat\_Body. Понятное дело, что к чему будем присвоено.



В данном ассете можно определять различные параметры, и, если у объекта включена физическая симуляция, то наш объект будет взаимодействовать с другими объектами согласно нашим настройкам. У материала есть специальный параметр Surface Type, позволяющий создать свой тип поверхности, и в зависимости от данного типа спаунить ножный нам эффект. Физические материалы могут немного отличаться по своим параметрам, но тип Surface у них может быть одним и тем же.



В нашем случае эти параметры не имеют для нас смысла, потому что мы используем данные физические материалы для определения поверхностей, а не физических симуляций.

Как можно присвоить наши материалы? Пол не является мешом, он является геометрией, поэтому ему присвоить физ. материал невозможно. А так в категории коллизии можно в параметре Phys Material Override выбрать нужный нам физический материал.

Изображение выглядит как текст

Автоматически созданное описание

Кубику присвоили PhysMat\_Ground.

Чтобы настроить физический материал на коллизиях персонажа, переходим в его меш во вкладку Physics:

Изображение выглядит как текст, снимок экрана, дисплей, компьютер

Автоматически созданное описание

В меше заданы коллизии, в которые мы попадаем при стрельбе. Всем им мы устанавливаем соответствующие физические материалы.

3. Как можно спаунить различные ниагара-системы в зависимости от данных физических материалов? Для того чтобы класс физического материала стал доступен в нашем проекте необходимо подключить дополнительный модуль PhysicsCore.

Далее в WeaponFXComponent переименуем Effect в DefaultEffect – данную ниагара-систему мы будем спаунить при попадании в объект, у которого не можем определить физический материал.

Так же нам необходимо каждому физическому материалу поставить в соответствие эффект, который мы хотим спаунить, когда в него попадаем, поэтому добавляем аналог STL-map UE под названием TMap. В качестве ключа будем хранить физический материал, а значения – ниагара-систему.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

У объекта HitResult имеется поле, которое содержит информацию о физическом материале, в который мы попали (данная переменная имеет тип weak-object (речь идет о weak\_ptr)).

Мы можем при нахождении нашего материала получить доступ к сырому указателю через функцию Get (прямо как у unique\_ptr).

4. Нам осталось добавить некоторые настройки в функцию ASTUBaseWeapon::MakeHit. Физический материал по умолчанию не передается в параметре HitResult (сделано для производительности), поэтому нам надо через CollisionParams явно указать, что мы хотим, чтобы физический материал попадал в итоговый объект HitResult:

Изображение выглядит как текст

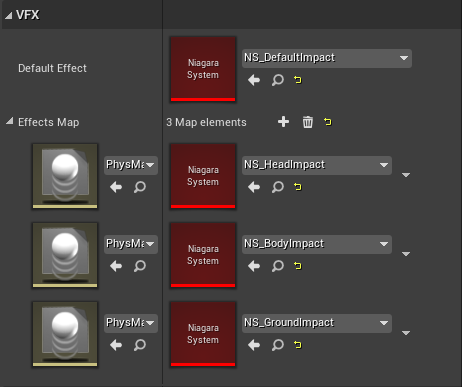
Автоматически созданное описание

А так же похожую настройку надо сделать для ASTUProjectile. Для него HitResult определяется непосредственно при столкновении коллизий, свойство, отвечающее за возврат физического материала, называется bReturnMaterialOnMove:

Изображение выглядит как текст

Автоматически созданное описание

5. Далее надо настроить в нашем WeaponFXComponent у RifleWeapon и Projectile наш словарь с физическими материалами и нужными ниагара-системами. Можно скопировать данные из одного компонента в другой (из Projectile в RifleWeapon).



6. Все-таки Projectile – это особый случай, сделаем для него одинаковый эффект для всех поверхностей. Создаем новую ниагара-систему NS\_ProjectileImpact, но уже через шаблон взрыва (встроенный). В нем надо так же поменять импульс на ось X. И устанавливаем ее в DefaultImpact, а словарь вообще никак не заполняем.