**9. Height drop health change**

1. Как можно двумя способами обработать падение персонажа? Какой мы выбрали?

2. Какая сигнатура у делегата, чем является аргумент, создали функцию, подписались на него (где)

3. Что нам для начала нужно получить при падении?

4. Создаем еще два проперти для обработки падения

5. Как запрогроммировали нанесения ущерба (понадобилась функция специальная)? Как мы можем специализировать поведение с помощью ивентов и структуры из делегата?

1. Перейдем в класс ACharacter и найдем там функцию Landed – она вызывается при контакте персонажа с поверхностью после падения.

В нашем классе BaseCharacter мы можем обработать событие падения несколькими способами: во-первых, можем переопределить виртуальную функцию Landed, либо подписаться на делегат LandedDelegate. Выберем второй вариант.

Изображение выглядит как текст

Автоматически созданное описание

2. Посмотрим сигнатуру нашего делегата, чтобы создать соответствующую функцию:



FHitResult – это специальная структура, в которой содержится информацию о контакте двух объектов – точка соприкосновения, нормаль и т.д. Мы об этом подробно поговорим далее.

Переходим в функцию BeginPlay нашего персонажа и подпишемся на делегат и создадим функцию:





Изображение выглядит как текст

Автоматически созданное описание

3. Мы имеем входную точку, когда наш персонаж соприкоснулся с поверхностью. Залогируем Z компоненту скорости (так как конечная скорость прямо пропорциональна высоте, с которой вы упали). Далее мы будем вычислять урон в зависимости от данной скорости (значение имеет отрицательный знак, потому что при падении вектор скорости противоположен Up-вектору).



4. Создадим еще два проперти, отвечающие за высоту падения и за наносимый урон при этом падении.

Изображение выглядит как текст

Автоматически созданное описание

Первый проперти отвечает за минимальную и максимальную скорость, при которой урон будет наноситься.

Второй – за сам урон.

В промежуточных значениях скорости будем высчитывать урон пропорционально.

5. Запрограммируем нанесения ущерба. Если значение скорости меньше минимального – просто выходим из функции. Если нет, то с помощью функции FMath::GetMappedRangeValueClamp() рассчитаем нужное значение урона.

Функция принимает три параметра, первый – входной отрезок, второй – выходной отрезок, третий – входное значение, относительно которого будет рассчитано результирующее значение относительно двуз диапазонов.

Останется только вызвать функцию TakeDamage:

Изображение выглядит как текст

Автоматически созданное описание

При этом, у DamageEvent находим PointDamageEvent, в котором содержится структура FHitResult. Таким образом мы можем создать PointDamageEvent и передать в данном ивенте данную структуру. И обрабатывать TakeDamage на основе ивента (для более сложной логики).