**5. Basic character movement (input)**

1. Сперва указываем разметку клавиш: два меппинга (в этот раз задавали по-особенному – несколько клавиш на одно действие), описать как

2. Как можно управлять поведение персонажа в зависимости от инпута (через что?), как мы будем делать в этом «билете», а что мы будем обрабатывать другим способом (в будущих)

3. Создали функции для привязки к нашим клавишам

4. Связываем наши функции с клавишами (все как обычно)

5. С помощью какой функции (встроенной) в функции перемещения (нашей) будем перемещать нашего персонажа, что принимает, какие есть функции (встроенные) для получения определенных векторов

6. Через клавишу F12 посмотрели реализацию используемой функции (встроенной, для перемещения), как устроена?

1. Для начала необходимо задать разметку клавиш, которыми мы будем управлять нашим персонажем. Так как мы хотим, чтобы зажатии клавиш наш персонаж перемещался, мы выбираем тип разметки Axis:



Стоит отметить, что мы можем размечать несколько клавиш на одно и то же действие (то есть на движение вперед у нас будет привязана как стрелочка вперед, так и «W»).

2. Управление поведением персонажа в зависимости от инпута можно настраивать как в контроллере, так и в самом персонаже. Контроллер имеет указатель на текущего пауна, которым он управляет, поэтому ввод можно проводить до пауна через контроллер. Мы начнем с персонажа, настроим базовый инпут в нем, но в дальнейшем будем использовать контроллер для различных неигровых событий (вызов паузы в игре, либо выход из игры).

3. Создаем функции, которые будут вызываться при нажатии наших клавиш. Функции будут принимать один параметр, в который будет передаваться состояние кнопки (1, 0, -1).

Изображение выглядит как текст, оранжевый

Автоматически созданное описание

4. Теперь нам необходимо связать функции MoveForward и MoveRight через PlayerInputComponent с разветкой клавиш, которую мы задали в редакторе UE:

Изображение выглядит как текст

Автоматически созданное описание

5. Для перемещения персонажа можно воспользоваться функцией AddMovementInput(), данная функция существует у каждого пауна.

Первый аргумент – направление перемещения персонажа (чаще всего используется unit-вектор (единичный вектор)). Так как мы хотим смещать нашего персонажа в направлении, в котором он ориентирован **в данный момент**, воспользуемся функцией, которая существует у каждого актора, GetActorForwardVector().

Более того, актор имеет свою **локальную систему координат**, помимо Forward-вектора у него есть Right-вектор и Up-вектор. Все эти вектора можно получить через соответствующие геттеры.

Второй параметр – это ScaleValue, число, на которое будет умножаться **покоординатно** наш вектор направления. Сюда мы передаем параметр, который приходит нам от кнопки.

Для второй кнопки используем аналогичную функцию, только с вызовом **GetActorRightVector**().

Изображение выглядит как текст

Автоматически созданное описание

6. Если мы через F12 посмотрим реализацию функции AddMovementInput(), то увидим, что данная функция является функцией пауна и оберткой над функцией AddInputVector() MovementComponent’a. Наш вектор направления умножается на второй параметр и передается в качестве аргумента в данную функцию, в которой уже осуществляется сама логика перемещения персонажа (изменение координаты в 3D-сцене).

Изображение выглядит как текст, снимок экрана, экран

Автоматически созданное описание