**14. EQS FIND HEALTH PICKUP**

1. Научим наших NPC пикапы здоровья через EQS. Создадим для этого новый EQS – как назвали, какой генератор добавили, что он делает? Как его настроили?

2. Добавляем тесты для нахождения подходящего пикапа. Пока что два. Один для распознавания за стеной он или нет – какой и как настроили? Второй – для нахождения ближайшего – какой и как настроили?

3. Нужно проверять, нужен ли пикап здоровья нашему NPC или нет. Создали для этого декоратор, от кого наследовались, определили две функции и проперти в ЗФ и их поведения в cpp – как?

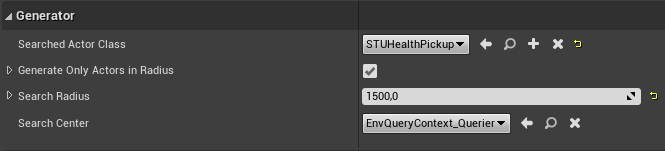
4. Забыли добавить в EQS еще один тест на случай.. на какой случай какой тест добавили? Добавляем наш декоратор на BT – кому и как настраиваем? Добавляем все это дело в основное дерево – куда, куда переносим сервис Fire?

1. Научим наших NPC пикапы здоровья через EQS. Создадим новый EQS\_FindHealthPickup. Для поиска пикапа здоровья воспользуемся специальным генератором Actors Of Class – он позволяет находить акторов определенного типа в мире.

Изображение выглядит как текст

Автоматически созданное описание

В панели Details определяем поле Searched Actor Class в STUHealthPickup. Search Radius в 1500 – таким образом наш EQS будет находить всех акторов STUHealthPickup в радиусе 1500.



Выберем для нашего EQS\_TestPawn новый EQS и увидим, что теперь наши найденные пикапы выделяются дебаг-сферами.

Изображение выглядит как пол, с плиткой, плитка

Автоматически созданное описание

2. Теперь с помощью тестов найдем подходящий пикап. Первый тест будет отвечать за видимость пикапа – если он находится за стенкой и персонаж его не видит, данный пикап нам не подходит. Этот тест называется Trace. Он выполняет трейс между всеми найденными акторами и пауном. Тест проходит, если трейс выполняется успешно – то есть по каналу Visibility (данный канал можно менять в настройках теста) существуют какие-то объекты между пикап-акторами и пауном.

Изображение выглядит как пол, земля

Автоматически созданное описание

Однако сейчас нам надо инвертировать условие Bool Match (из true в false), так как находятся сейчас те, с которыми трейс неудачный.

Изображение выглядит как пол, земля

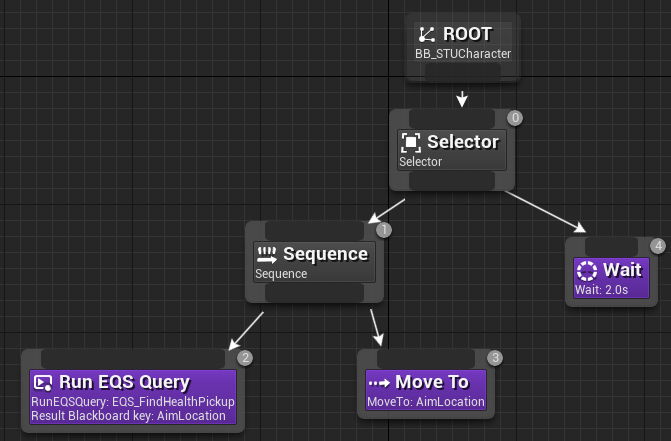
Автоматически созданное описание

Теперь надо найти самый ближайший к нам пикап. Для этого добавляем тест Distance и выбираем Filter Type Minimum и инвертировать Scoring Factor из 1 в -1.

Изображение выглядит как текст, электроника, снимок экрана

Автоматически созданное описание

И запустим тест нашего EQS изменив дерево поведения следующим образом:



Важно, чтобы и там, и там были AimLocation установлены. Для того, чтобы иметь возможность выйти из игры и мочь перемещать предметы, нужно нажать клавишу Eject.

3. Таким образом, мы можем добавить данную последовательность в наше дерево поведения, однако нужно проверять, нужен ли данный пикап здоровья нашему NPC. Для этого мы создадим специальный декоратор, который будет вычислять данное условие.

Создадим собственный C++ декоратор, назовем STUHealthPercentDecorator, наследуемся от класса BTDecorator.

Добавляем **конструктор** и **функцию** CalculateRawConditionValue из базового класса, в которой будем вычислять необходимое нам условие, по которому будет определяться то, будет секвенция под нашим декоратором выполняться или нет, а так же **проперти** допустимого процента здоровья персонажа, ниже которого он начинает искать наш пикап.

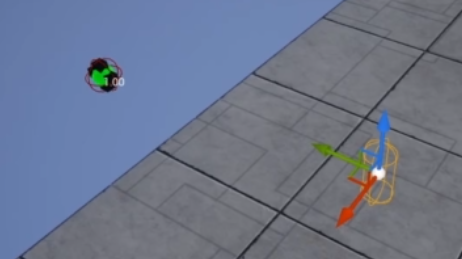
Изображение выглядит как текст

Автоматически созданное описание

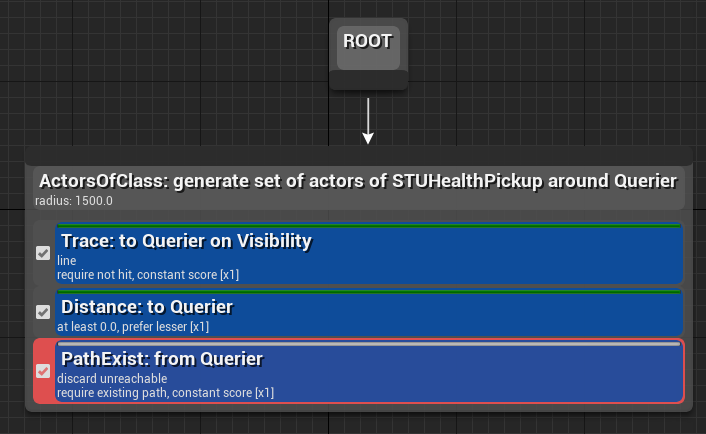
Изображение выглядит как текст

Автоматически созданное описание

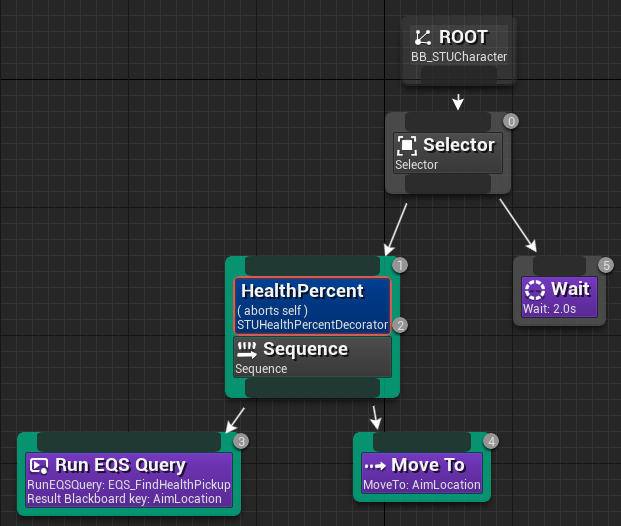
4. Забыли добавить в EQS тест на случай, если вдруг у нас наш HealthPickup оказался вне нашего навигационного меша.



Тесты проходят, но NPC не сможет попасть в данную точку. Для исправления этого поведения добавим дополнительный тест в EQS, который будет определять, существует или нет навигационный путь в найденную точку. Называется он Pathfinding. С настройками по умолчанию он отсекает все точки, до которых путь не существует (проперти Test Mode = Path Exist).



Переходим в BT и добавляем нашей Sequence наш декоратор, у него необходимо поставить проперти Observer aborts в Self.



Таким образом, персонаж будет искать пикап только в том случае, если его здоровья <= установленного в декораторе значения.

Добавим нашу секвенцию поиска пикапа здоровья в наше основное дерево в самый левый край (она будет иметь наибольший приоритет) и перенесем сервис стрельбы из таска Attack в ноду Selector, чтобы у нас персонаж стрелял, даже когда бежит за пикапом.

