**4. Move AICharacter to random point. Task**

1. Создадим свой собственный таск – что будет делать?

2. Как создать ассет атска в BP, как мы сделаем? От кого наследуемся? Что нужно подключить в Build.cs?

3. Что объявляем в ЗФ нашего таска? Два проперти и две функции. Для чего конструктор? Когда вызывается вторая функция? Какой тип имеет второе проперти?

4. В cpp-файле добавляем 3 ЗФ, в конструкторе что делаем? А что в функции? Что она возвращает? Подсказка: как получить контроллер, блекборд, пауна, систему навигации, рандомно сгенерированную позицию и установить полученное значение в блекборд?

5. Что теперь делаем в UE у блекборда, дерева поведения и AI-контроллера?

6. Какая есть нода для теста дерева поведения?

1. Создадим свой собственный таск, который будет вычислять рандомную локацию и NPC будет патрулировать сцену, передвигаясь между сгенерированными точками.

2. Чтобы создать ассет таска в блюпринте достаточно нажать New Task в Behavior Tree. Мы создадим его через C++.

Базовый класс для таска называется BTTaskNode, от него наследуемся и называем класс STUNextLocationTask в поддиректории AI/Taks.

В Build.cs надо добавить нашу поддиректорию, а так же нам надо добавить два модуля для работы с AI: GameplayTasks и NavigationSystem.

3. В ЗФ нашего таска объявляем конструктор (для косметики, там объявим имя ноды для отображения в UE). Провалившись в ЗФ базового класса находим функцию ExecuteTask – она вызывается, когда начинается выполнение нашего таска. Так же нам понадобятся два проперти – первое для радиуса, в котором мы будем вычислять рандомную точку вокруг нашего персонажа, а второе будет хранить Blackboard Key, в которую мы будем записывать вычисленную рандомную точку. Отметим, что FBlackboardKeySelector – это структура (далее это увидим).

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

4. В cpp-файле добавляем 3 ЗФ. В конструкторе переопределяем переменную из базового класса NodeName.

Изображение выглядит как текст

Автоматически созданное описание

В функции будем вычислять радномную точку вокруг персонажа и записывать в Blackboard. Controller – указатель на контроллер NPC, Blackboard – указатель на текущую доску, Pawn – указатель на Pawn, NavSys – указатель на NavigationSystem (для нахождения рандомной точки).

Функция NavigationSystem, возвращающая рандомную точку, вычисляет на навигационном меше рандомную точку, куда наш персонаж может попасть – под капотом у нее алгоритм поиска пути, поэтому если она нашла точку, значит мы 100% уверены, что из текущей точки в ту, которую она нашла, можно попасть.

У нее несколько параметров – центр, относительно которого ищем, радиус поиска, в третий передается найденная точка (FNavLocation).

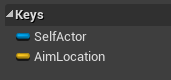
Так же данная функция возвращает true или false, поэтому эту проверку тоже организовали – если не найдена – failed, если найдена – устанавливаем ее в Blackboard – для этого вызываем функцию SetValueAsVector, которая принимает имя ключа (его получаем из проперти для структуры) и значение (получаем его из поля Location структуры FNavLocation).

Функцию возвращает элемент из enum EBTNodeResult:

Изображение выглядит как текст

Автоматически созданное описание

5. Теперь в Blackboard определим нашу AimLocation, удалим остальные. Так же удаляем таски кроме MoveTo в BehaviorTree и добавляем наш новый таск. Из ивент графа контроллера убираем все, кроме RunBehaviorTree.



Изображение выглядит как текст, сиреневый

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

6. Протестировать работу статуса можно с помощью ноды FinishWithResult. Если он возвращает Succeeded, то все работает, если выберем в его настройках Failed, то MoveTo не будет вызываться.

