Что такое искусственный интеллект?

Искусственный интеллект — свойство интеллектуальных систем выполнять творческие функции, которые традиционно считаются прерогативой человека; наука и технология создания интеллектуальных машин, особенно интеллектуальных компьютерных программ.

Как работает ИИ?

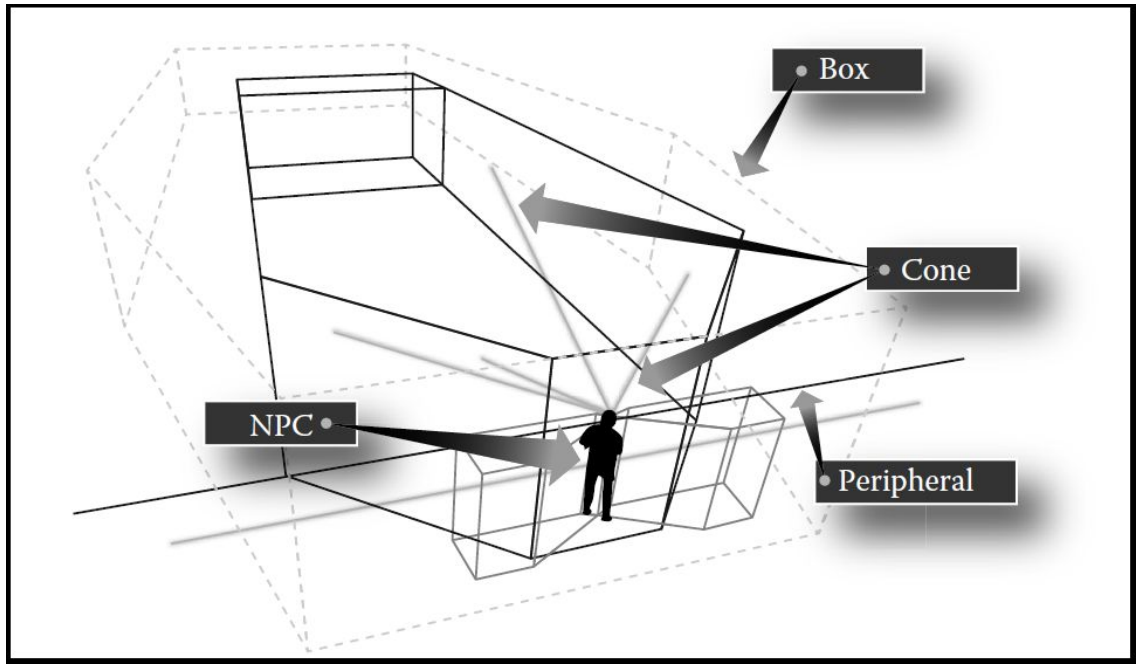
ИИ в играх — это набор алгоритмов, которые диктуют поведение NPC в разных ситуациях. Игровой ИИ неспособен на мышление или творчество, его действия предопределены разработчиками. Несмотря на такие ограничения, грамотно созданный ИИ подстраивается под ситуацию и меняет поведение в зависимости от контекста. В современных играх используются разные подходы для создания ИИ. В основе лежит общий принцип: получение информации → анализ — действие. Под катом — самые популярные методы и примеры использования ИИ в играх.

-Как игровой ИИ получает информацию

Получение информации происходит примерно так же, как и в реальном мире — у ИИ есть специальные сенсоры, при помощи которых он исследует окружение и следит за происходящим. Сенсоры бывают совершенно разными. Это может быть традиционный конус зрения, «уши», которые улавливают громкие звуки, или даже обонятельные рецепторы. Конечно, такие сенсоры — всего лишь имитация реальных органов чувств, которая позволяет сделать игровые ситуации более правдоподобными и интересными.

Виртуальные рецепторы устроены по-разному. В Metal Gear Solid у противников совсем простой конус зрения, отсутствует периферийное зрение, поэтому они видят только то, что происходит прямо перед ними.

Комплексный визуальный сенсор. У врагов есть основная зона видимости в форме вытянутого шестиугольника. Здесь моб видит лучше всего, поэтому в это пространство вообще лучше не заходить. Есть и более крупный шестиугольник, который имитирует периферийное зрение — там противник видит хуже. В такую же зону входят удалённые участки, которые недостаточно хорошо просматриваются.



СДЕЛАНО

-Как ИИ принимает решение

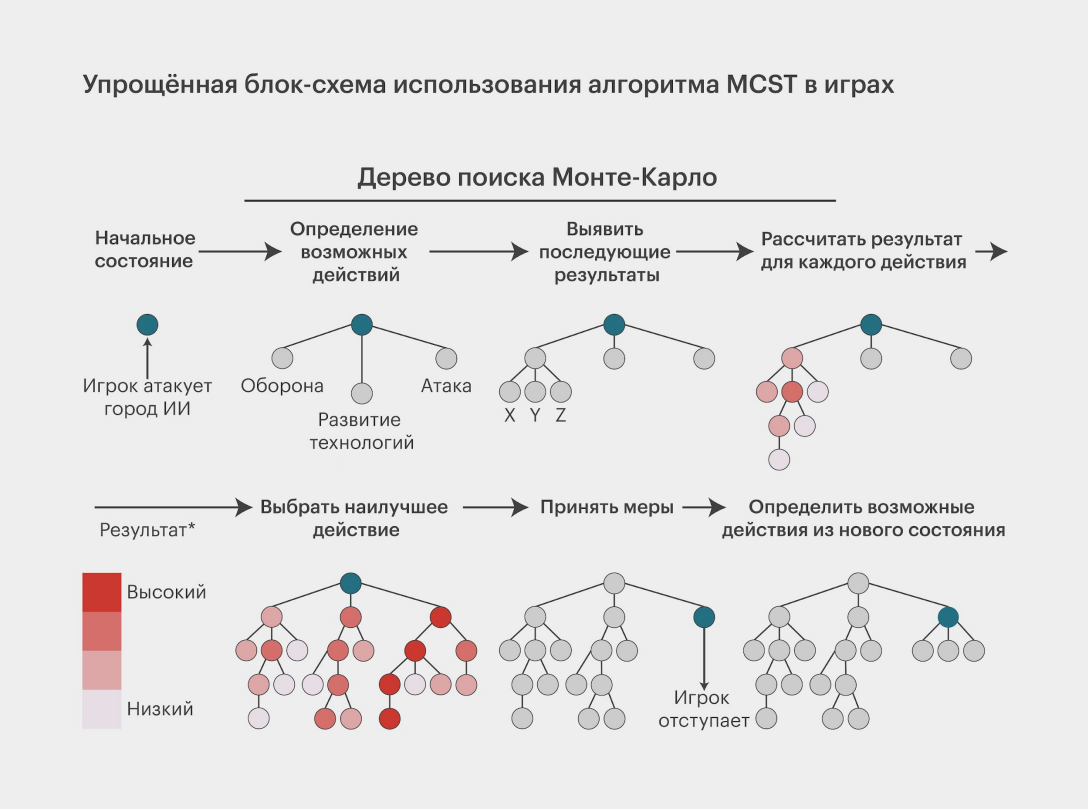
Когда ИИ получил информацию, он начинает «обдумывать» свои действия, анализируя обстановку. Обычно в этом участвует сразу несколько систем ИИ, отвечающих за разные вещи. Часто разработчики добавляют подобие коллективного интеллекта, который следит за тем, чтобы действия отдельных агентов не противоречили и не мешали друг другу. При этом сами мобы зачастую даже не знают о существовании своих союзников — эта информация им не нужна, потому что за координирование действий отвечает ИИ более высокого уровня.

- В играх есть несколько подходов, которые чаще всего используются для принятия решения. Один из самых простых и понятных подходов — это **rule-based ИИ**. В основе лежит список правил и условий, заранее созданный разработчиками.
- Следующий распространённый способ принятия решений — конечные автоматы. Этот подход позволяет NPC беспроблемно переходить между разными состояниями.
- Дерево поведения — это более формализованный подход построения поведения мобов. Его особенность заключается в том, что все состояния персонажа организованы в виде ветвящейся структуры с понятной иерархией. Дерево поведения содержит в себе все возможные состояния, в которых может оказаться моб. Когда в игре происходит какое-то событие, ИИ проверяет, в каких условиях находится NPC, и перебирает все состояния в поисках того, которое подойдёт для нынешней ситуации.

Один из широко используемых алгоритмов называется конечным автоматом (FSM или finite-state machine). Его ввели в разработку видеоигр в 1990-х годах. В FSM-алгоритме разработчик обобщает все возможные ситуации, с которыми может столкнуться ИИ, а затем программирует конкретную реакцию для каждой из них. Например, в шутерах искусственный интеллект атакует, когда появляется игрок, а затем отступает, когда его собственный уровень здоровья становится слишком низким.

В примере алгоритма FSM NPC может выполнять четыре основных действия в ответ на возможные ситуации: поиск помощи, уклонение, блуждание и нападение.

Более продвинутый метод, который используют разработчики для повышения персонализированного игрового опыта, — алгоритм дерева поиска Монте-Карло (MCTS или Monte Carlo Tree Search). Алгоритм MCTS был создан для предотвращения аспекта повторяемости, который присутствует в FSM-алгоритме. MCTS-алгоритм сначала обрабатывает все возможные ходы, доступные NPC в конкретный момент времени. Затем для каждого из этих возможных ходов он анализирует все действия, которыми игрок мог бы ответить. А далее — снова возвращается к оценке NPC уже на основе информации о поступках игрока.



На каком языке написать ИИ для НПС?

- C#

Если вы слышали про метавселенные, виар и дополненную реальность, то C# — один из языков, на котором можно делать все эти штуки. Смысл в том, что в большинстве случаев там нужна 3D-графика, для которой можно использовать движок Unity. А C# как раз отлично дружит с Unity и позволяет программировать и управлять логикой внутри метавселенных и дополненной реальностью.

- C++

На данном языке написано большинство движков для создания игр. Его возможности программирования на низком уровне предлагают высокую степень гибкости, которая просто недоступна для языков программирования игр более высокого уровня, таких как Python и C #. Благодаря своей гибкости и сходству с машинным кодом C ++ отлично подходит для оптимизации производительности, что имеет решающее значение в контексте игрового процесса.

Компании использующие ИИ в играх

Epic Games — использует движок Unreal Engine, по умолчанию поддерживает сразу два языка программирования: текстовый C++, в котором нужно писать строчки кода, и визуальный язык Blueprints, в котором игровая логика выстраивается при помощи связанных между собой блоков.

Игры Ubisoft выпускаются на собственных движках – Anvil, Dunia Engine, Disrupt и Snowdrop. Они построены на языке C++.

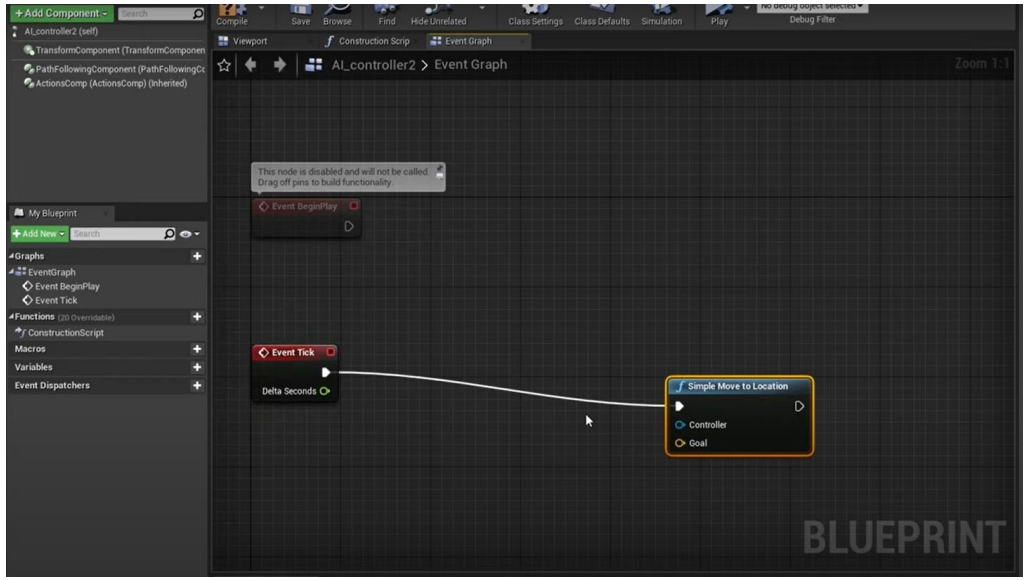
RockStar Games – использует Rockstar Advanced Game Engine (часто сокращенно RAGE) - это проприетарный игровой движок, разработанный RAGE Technology Group, подразделением студии Rockstar Games. Движок построен на C ++.

Преимущества использования C++ для разработки игр

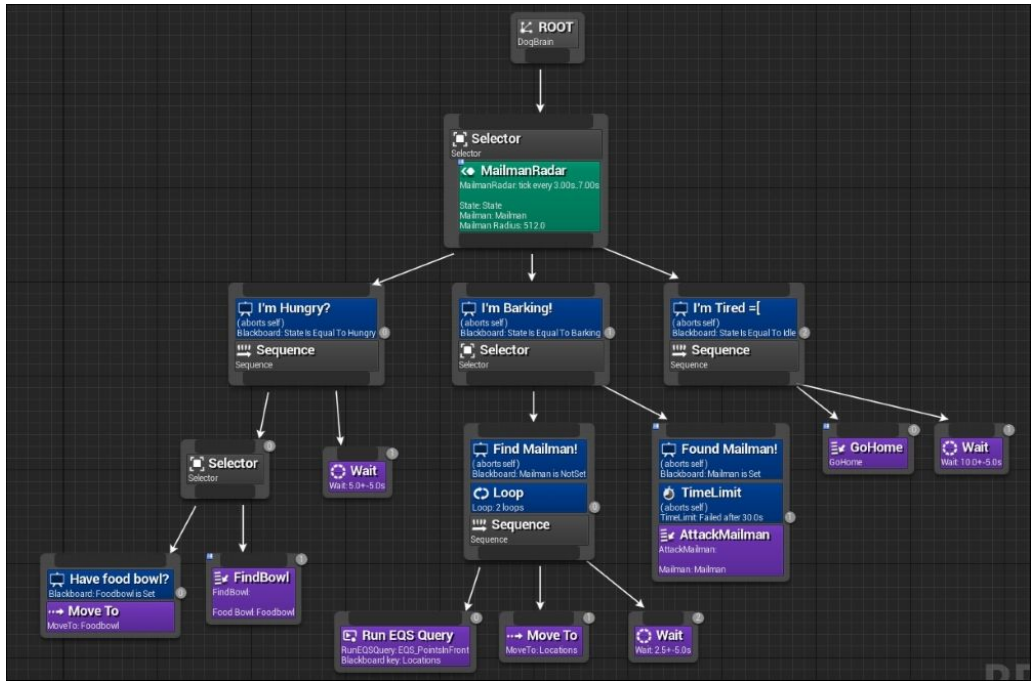
Самая главная причина выбора «плюсов» – Unreal Engine 4 (UE4) написан на них. Знание его фундамента позволит максимально эффективно использовать возможности движка. Помимо этого у C++ есть немало возможностей для геймдева. Это компилируемый язык с кучей всевозможных настроек, поддержкой ООП и управлением мельчайшими деталями кода.

Особенности создания ИИ в Unreal Engine

ИИ можно создать как и любую программу на языке C++ или визуально с помощью Blueprint. Это отлично подойдет для простых задач, но для более сложных используется специальный инструментарий - дерево поведения (Behavior Tree).



- **Blueprints** — это система визуального скриптинга Unreal Engine. Вместо построчного написания кода всё можно делать визуально: перетаскивать ноды (узлы), задавать их свойства в интерфейсе и соединять их «провода».
- **Behavior Tree** - это ориентированный ациклический граф, узлами которого являются возможные варианты поведения робота.



Такие инструкции, как поступать в той или иной ситуации, формируют дерево поведения (нейронную сеть), которая в свою очередь объединяет задачи (нейроны), логические связи, цепочки. Такими задачами могут быть алгоритмы на передвижение, атаку, взаимодействие с игроком и другие.

Как создать ИИ в Unreal Engine?

Чтобы создать ИИ-персонажа, понадобятся две вещи:

1. Тело — это физическая оболочка персонажа.
2. Мозг — AI Controller - это то, каким образом ИИ принимает решения. Для его настройки можно использовать несколько способов, в том числе код C ++, Blueprints или деревья поведения.

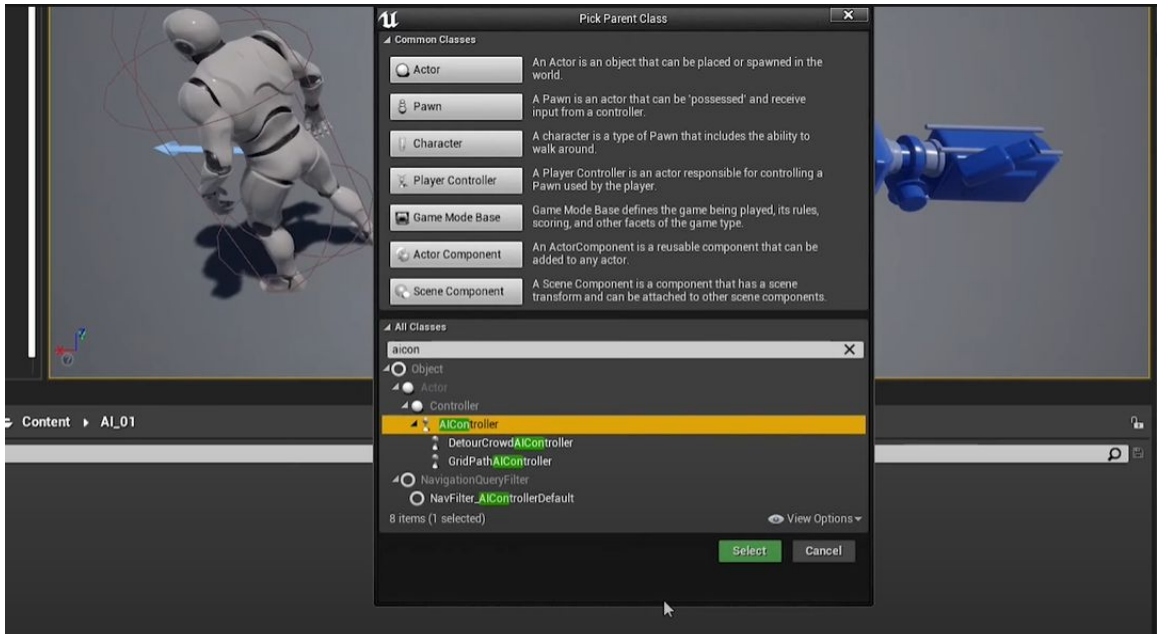
AI Controller (Artificial Intellegence – «искусственный интеллект ») – **это** нефизический объект, который может управлять персонажем. В нем предварительно прописаны инструкции для NPC ботов: как реагировать на окружающий мир.

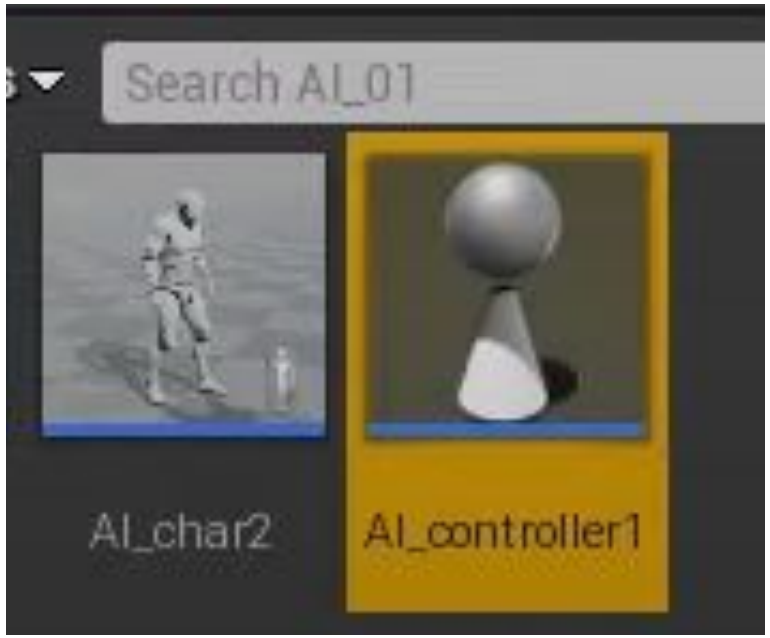
Тело

В Unreal Engine за тело можно взять готовый шаблон анимированного персонажа от третьего лица. Затем его нужно дублировать и удалить всю логику на управление.

Мозг

Для мозга используем AI Controller. Он находится во вкладке «Blueprint Class».

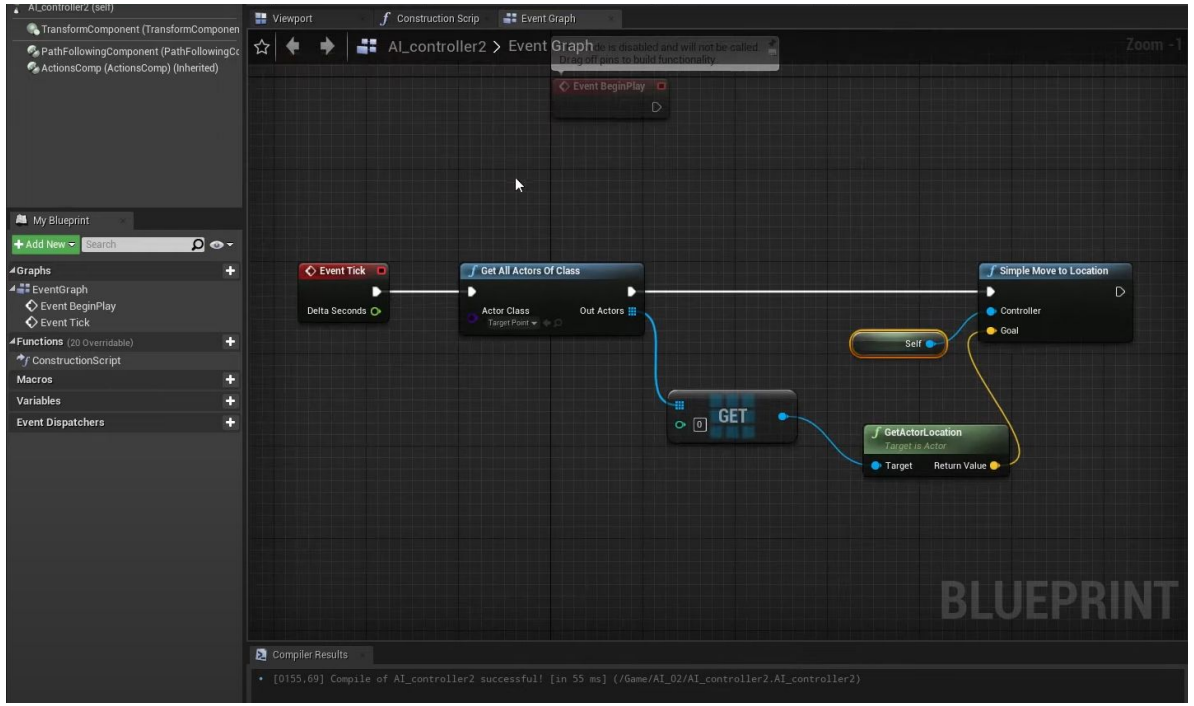




AI Controller

Прикрепим логику на «Event Tick», соединим с ссылкой на объект (Get All Actors Of Class), к которой наш бот будет бежать. Затем с помощью функции «Get» получим ссылку на точку и присоединим «Get Actor Location», которая даёт координаты точки.

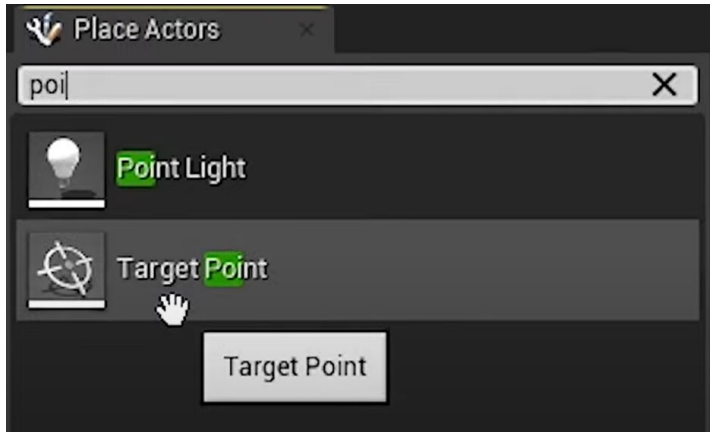
В контекстном меню выберем нод «Simple Move To Location» и прикрепим его на «Get Actor Location». Также подключим ссылку на AI Controller и в «Goal» координаты, к которым нужно двигаться.

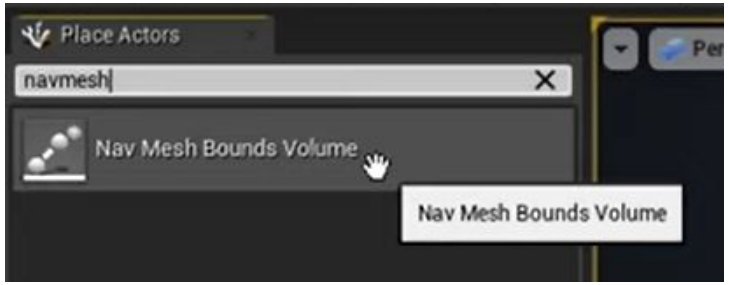


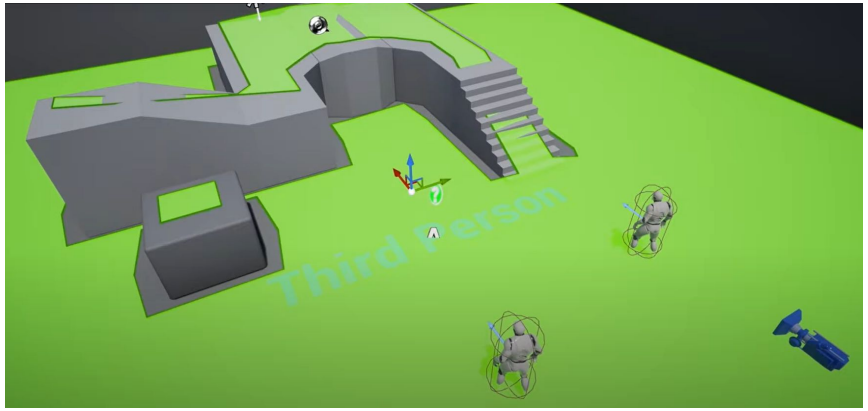
Создание точки и навигации ИИ.

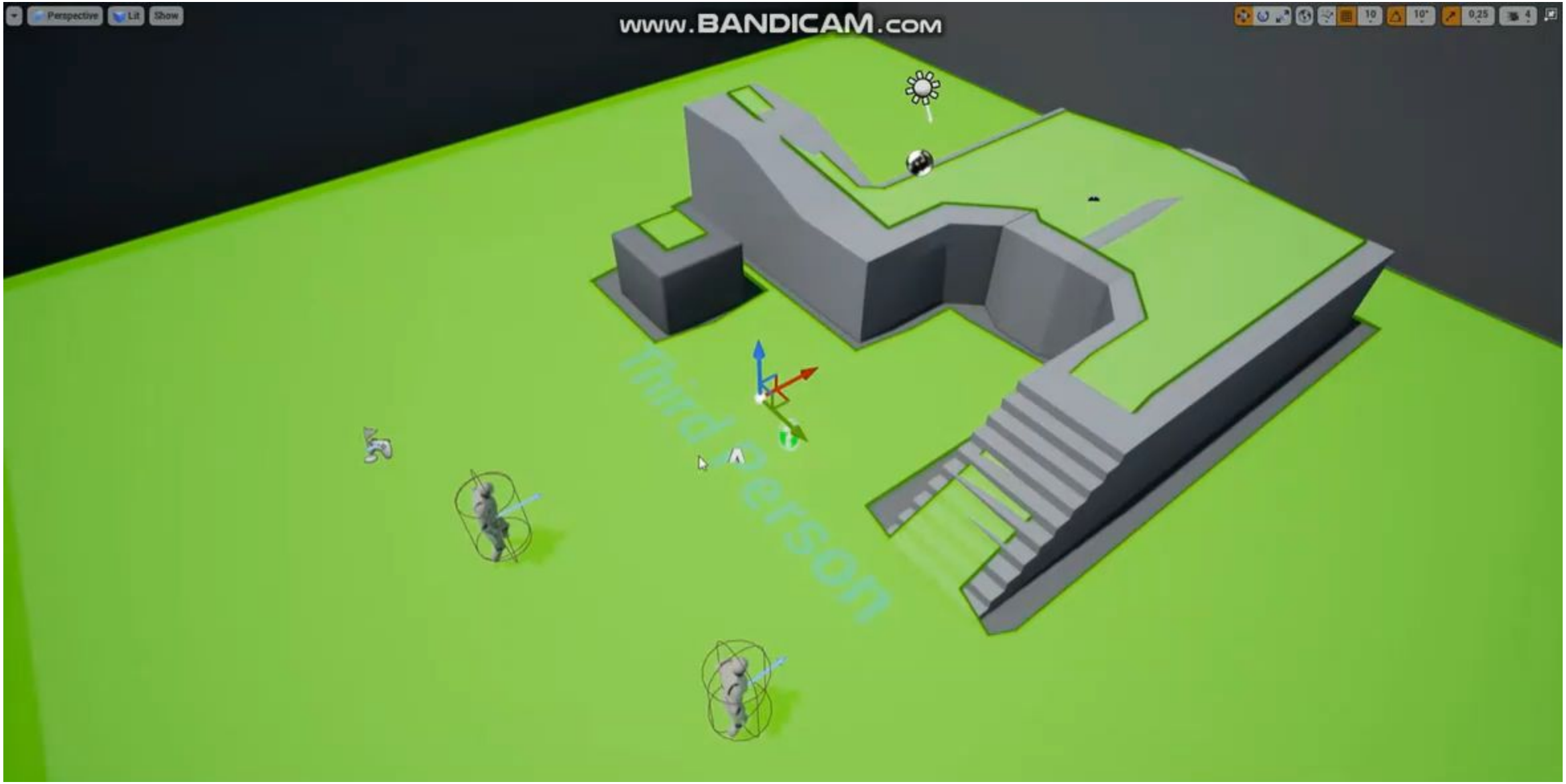
В панели «Place Actors» возьмём объект «Target Point» (именно к нему наш бот будет бежать) и выставим его на сцену.

Затем выберем меш «Nav Mesh Bounds Volue », он заранее просчитывает пути для ИИ.








Perspective
Lit
Show
www.BANDICAM.com
third person

Будущее искусственного интеллекта в играх

Искусственный интеллект в играх значительно повзрослел за последнее десятилетие. Создание эффективных систем ИИ теперь стало таким же важным для разработчиков игр, как создание надежного игрового процесса и ярких визуальных эффектов. Студии начали назначать специальные команды программистов для разработки ИИ с самого начала цикла разработки игры, тратя больше времени и ресурсов на попытки создать разнообразных, способных и последовательных неигровых персонажей (NPC).

Таким образом, на сегодняшний день главной целью программистов является: сделать систему ИИ максимально человечной, способной принимать собственные решения, понимая, что это сделает игру еще более увлекательной для игроков.