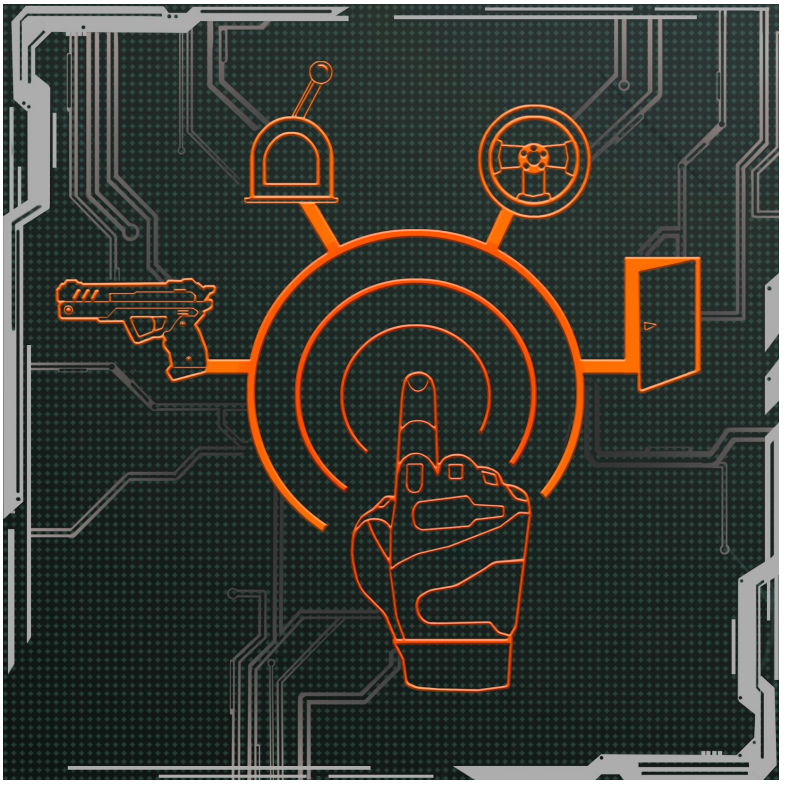
Character Interaction

Документация

(Unreal Engine 4 ассет)

https://www.unrealengine.com/marketplace/character-interaction



Видео демонстрации проекта
Видео первого обновления (IK, Поддержка вида от первого лица и т.д.)
Видео второго обновления (Расширенная система перемещения)
Видео третьего обновления (полная переработка проекта)

Документация написана автором проекта ZzGERTzZ.

Manual version : 3.0
Last update : 12/07/2018

1. Введение:

Проект **Character Interaction** создан с целью предоставить разработчикам заготовку для игр различных жанров включающие в себя систему перемещения персонажа, взаимодействия с различными объектами, использование разнообразного огнестрельного оружия, использование транспортных средств, подбор и создание различных предметов, создание различных ИИ врагов и т.д.

В пакет входят следующие необходимые элементы для геймплея:

Базовое перемещение с процедурными наклонами, чтобы можно было обходится меньшим кол-вом анимации и при этом чувствовалось реакция персонажа на смену направления при движении. Так же полностью реализовано обращение с оружием включая прицеливание, перезарядку, стрельбу в любом направлении, экипировка и смена оружия. Присутствует система прыжков, движения сидя, перемещение во время прицеливания и прочее…

Взаимодействия с объектами. Это могут быть как различные переключатели, рычаги, кнопки так и вентили или любые другие интерактивные объекты ограниченные лишь фантазией разработчика. В проекте изначально представлены 10 заготовок таких объектов (могут расширяться в обновлениях). Это различные кнопки, мебель, рычаги, стенка для перелезания... Ознакомится можно в видеороликах в заголовке.

Подбор предметов. Подбор предметов является неотъемлемой частью жанра. В классе написано автоопределение положения персонажа и выбор нужной анимации, поэтому достаточно просто указать имя предмета его модель и текст. Также учтен подбор специфических предметов.На примере демо уровня это пистолет который надо подбирать определенной анимацией. На примере уровня “ZombieCity” это бронежилет увеличивающий здоровье персонажа или газовая маска позволяющая проходить сквозь загазованные комнаты.

Инвентарь. В проекте присутствует класс инвентаря при помощи которого вы можете создавать различные квестовые предметы, переносить ресурсы, также есть поддержка типов патрон в инвентаре для исопльзования их разными оружиями и прочее.

Огнестрельное Оружие. Полнофункциональный класс оружия содержит в себе настройки для создания различных огнестрельных оружий. Включает себя баллистику (можно использовать летящие снаряды или трейсовую систему),перезарядку, отдачу и разброс, физические импульсы, гильзу и обойму, анимации как оружие так и персонажа обладающего данным оружием... В ролике присутствует перечень настроек. Настройки были значительно расширены в 3ем обновлении.

Склад оружия. Данный класс содержит в себе систему создания и хранения различного оружия. Позволяет в два клика создать новое оружие и добавить его в игру.

Менеджер оружия. Присутствует менеджер оружия в нем визуально представлено наличие вашего оружия как огнестрельного так и для ближнего боя. Также через данный менеджер вы можете снимать и одевать оружие на персонажа или выкидывать его и подбирать обратно.

ИИ. Базовый искусственный интеллект, который может видеть, слышать, преследовать и атаковать игрока как в ближнем бою так и при помощи огнестрельного оружия. Присутствуют несколько примеров такие, как солдат, зомби(медленный), кибер зомби (быстрый). Персонажи полностью про анимированы, настроены и озвучены. Так же как пример работы с ии для каждого класса добавлены свои особенности, у солдата использование шлема, у зомби рандомизатор персонажа и т.д.

Транспорт. В проекте присутствует класс позволяющий создавать различный транспорт. Класс поддерживает взаимодействие с персонажем, а также обширное кол-во модулей таких как: декор модуль (перекраска машины - использование аэрографии),нитро(ускорение), расход топлива, импульсный оборонный модуль, система вооружения и прочее. В проекте представлены два типа автомобилей: базовый автомобиль и боевой автомобиль. Машины полностью проанимированы, настроены и озвучены.

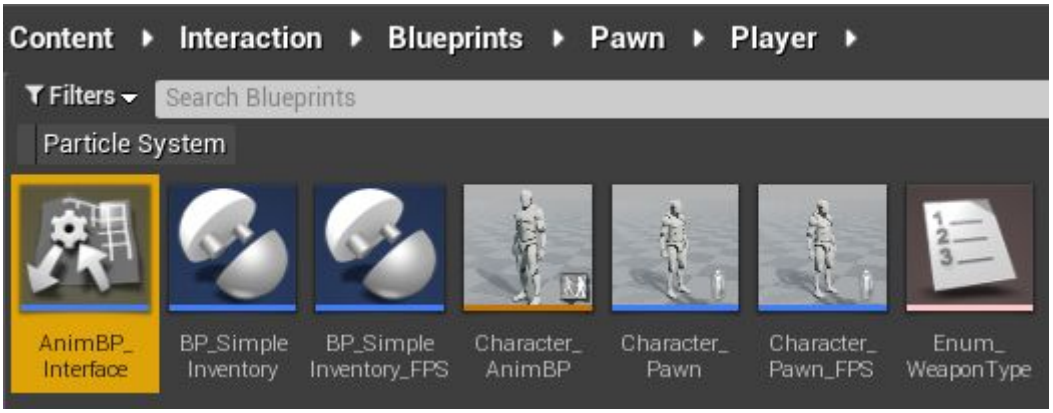
Ближний бой. Присутствует базовый ближний бой. Присутствует функционал настройки ударов персонажа взависимости или нет от оружия в руках (к примеру удары прикладами огнестрельного оружия). Также можно добавлять различное оружие для ближнего боя (по умолчанию в виде примера добавлен нож). Присутствует система добивания врагов - некие захваты, присутствует функционал настроек анимации, условий для использования, звуков, эффектов и прочее.

Также в пакете присутствуют различные звуки, модели оружия, полностью физическая мишень для стрельбы, HUD для персонажа, HUD для автомобиля, модели игрока включая дополнительные обвесы (бронежилет, газовая маска, кобура для пистолета, механическая кобура для двуручного оружия), модель солдата и шлем для него, 3 разделенных для рандомизатора персонажа зомби,
3 разделенных для рандомизатора персонажа киберзомби. Демонстрационные уровни различных классов, включая полностью геймплейный уровень город зомби.

В коде проекта содержится большое кол-во комментариев, а также интерактивные туториалы. Присутствует поддержка первого и третьего лица.
Надеюсь этот стартовый пакет окажется полезным разработчикам в создании своих игр и упростит начало разработки.

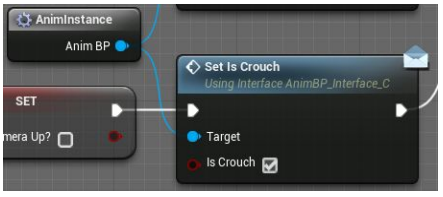
2. Базовое перемещение :

Перемещение персонажа основано на базовом классе перемещения от “*Epic games*” **character movement**. Всвязи с этим все параметры перемещения настраивайте в данном компоненте. Персонаж умеет двигаться в любом направлении, перемещаться сидя, совершать прыжки, перемещаться при прицеливании с оружием в руках, использовать ближний бой, машины и прочее.
Анимация от пешки в anim blueprint передается посредством интерфейсов

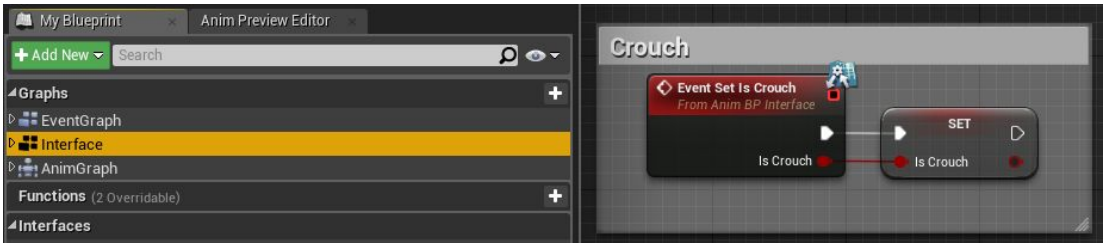


Это сделано ради того чтобы не привязывать пешку персонажа к определенному anim blueprint и впоследствии упрощая интеграцию других персонажей, которые используют свой скелет. После интеграции персонажа вы можете использовать прямую ссылку на anim blueprint посредством “cast to” или же добавляя функции в интерфейс “AnimBP_Interface”.

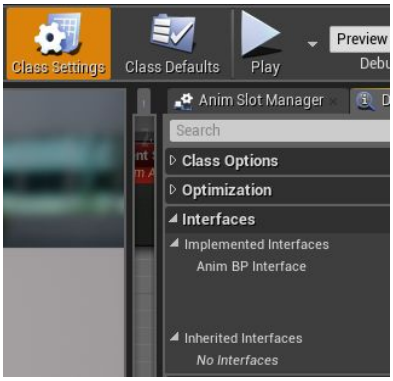
Разберем как это выглядит на примере приседания. После нажатия на кнопку приседания идут различные просчеты (может ли персонаж присесть в текущий момент?) и если персонаж приседает то мы вызываем сообщение интерфейса под названием Set Is crouch



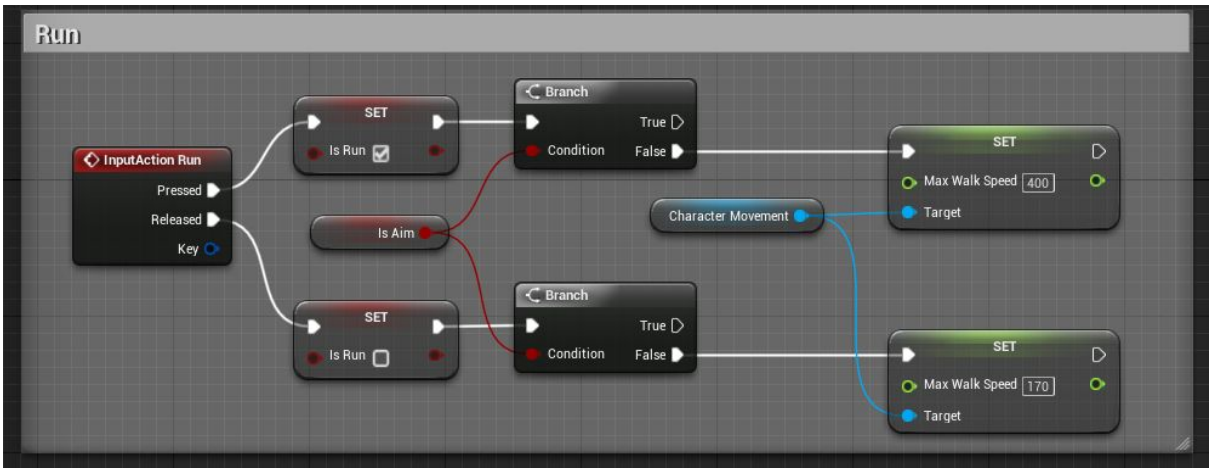
(это функция которую мы предварительно создали в AnimBP_Interface) для вызова в AnimBlueprint необходимого ивента который активирует анимацию персонажа и переводит его в состояния приседа.
Все ивенты интерфейса AnimBP_Interface в anim blueprint располагаются здесь



Чтобы вызвать ивенты интерфейса вы должны предварительно его подключить в “Class settings”.



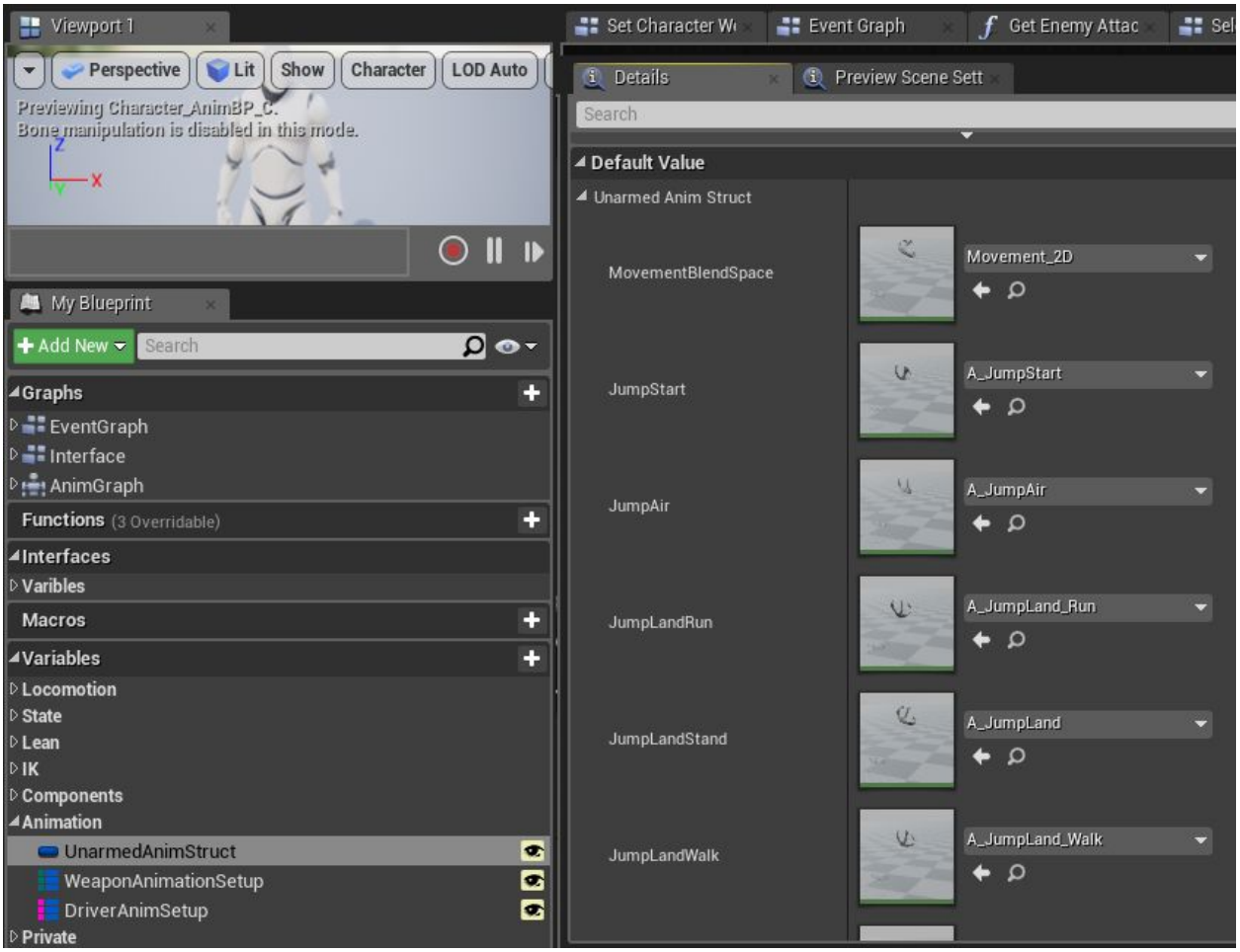
Что касается логики перемещения персонажа то оно построено на динамическом изменении параметров класса компонента “character movement” от “epic games”. На примере бега меняется максимальная скорость при нажатии на клавишу ускорения (по умолчанию shift)



В зависимости от скорости персонажа меняется и анимация в anim blueprint.
Таким образом осуществляется связь анимации и перемещения персонажа в пакете “character interaction”.

Настройка анимации

В проект добавлены структуры для анимации для удобного добавления или замены анимации персонажа. Чтобы изменить анимацию откройте “**Character_AnimBP**” и найдите категорию “**Animation**” в списке переменных.



Выделяя различные структуры анимаций вы можете заменить текущую анимацию у персонажа, добавить новые типы анимаций: перемещение с оружием или поведение в различных транспортных средствах.

3. Взаимодействия с объектами :

Класс взаимодействии с объектами служит для визуального представления взаимодействия персонажа с окружающей средой. Данный класс разработан таким образом, чтобы разработчики могли внедрять различной сложности объекты взаимодействия, изначально в проекте присутствует 10 типов объектов и анимация к ним - это различные рычаги, кнопки, мебель, вентиль и прочее. С демонстрацией можно ознакомится в видеоролике.

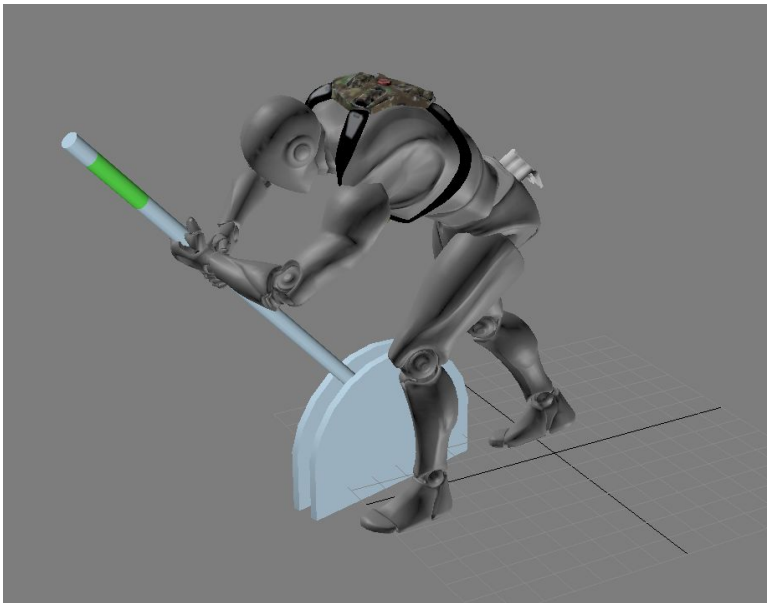
Как это работает:
Система взаимодействия работает по следующей схеме - в акторе “BP_InteractionObject” присутствует триггер при пересечении которого персонаж может начать взаимодействие с объектом посредством нажатия кнопки взаимодействия (по умолчанию “E”). После нажатия кнопки происходит просчет в пешке - может ли в данный момент персонаж взаимодействовать с объектом (не стреляет ли он? не взаимодействует ли он уже с объектом в данный момент? и так далее) в случае если персонаж может взаимодействовать с объектом то BP_InteractionObject запускается логика, которая подбирает необходимые анимации, плавно меняет расположение персонажа, учитывает тип объекта… и впоследствии по заданному таймеру вызывает диспатчер результата по которому вы уже выбираете различные ивенты (открытие двери, включения электричества и прочее).

Как создать свой объект для взаимодействия:

Разберем первый пример с рычагом из демо уровня, который выдвигает мост.

***Анимация**:*

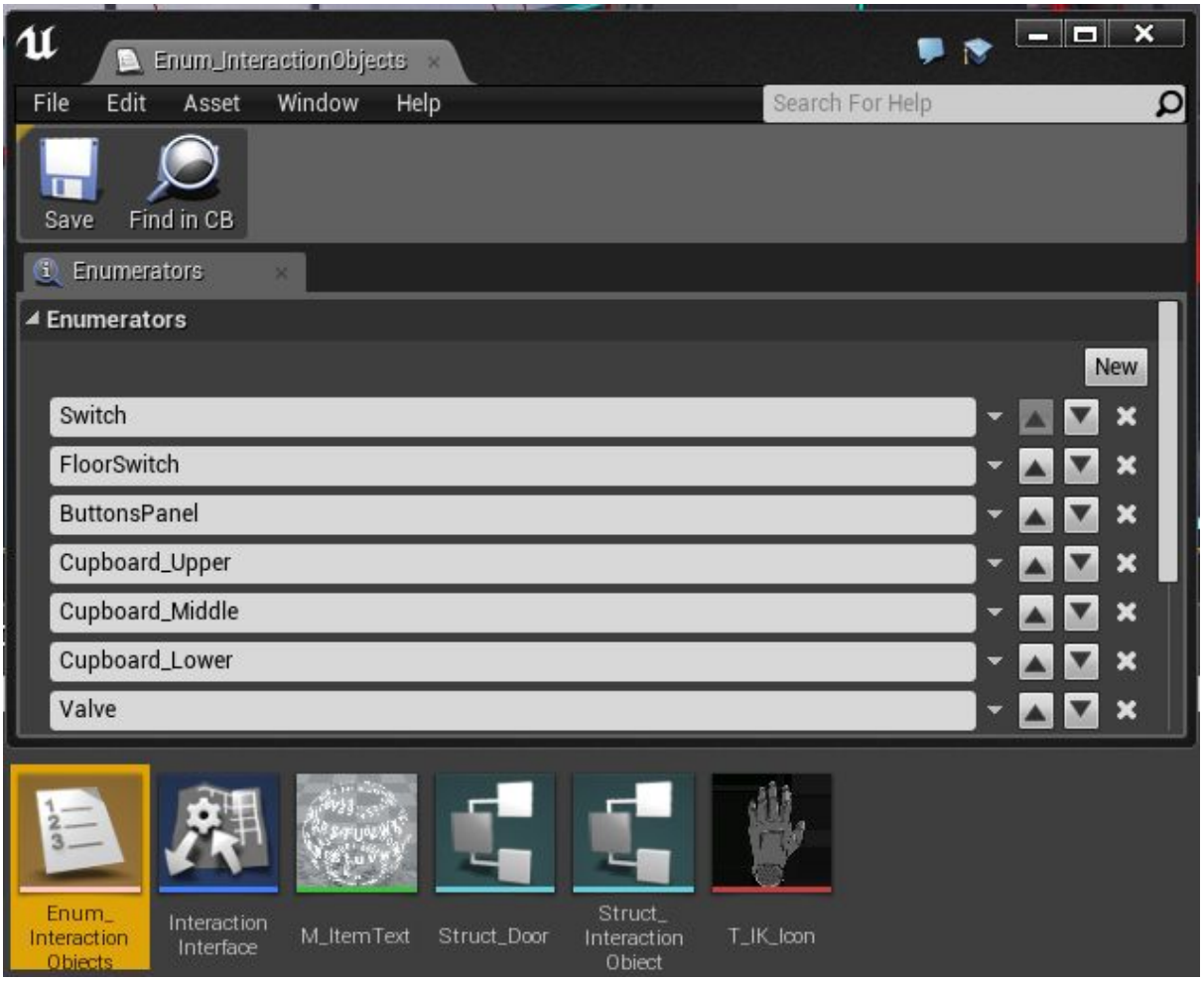
Первым делом вам нужны необходимые анимации, как персонажа так и объекта в данном случае рычага. Чтобы автоматом подхватить верное расположение персонажа и объекта создавайте анимации исходя из нулевых координат. Другими словами анимируете объект взаимодействия относительно персонажа как указано на картинке.



Добавления нового объекта:

После создания анимации и добавления ее в unreal engine 4, добавим наш новый объект в список объектов для многоразового использования (если в планах использовать объект только один раз и на одном уровне - достаточно добавить его в массив в BP_InteractionObject на сцене, но я рекомендую добавить его в список, как это сделано по умолчанию со всеми объектами).

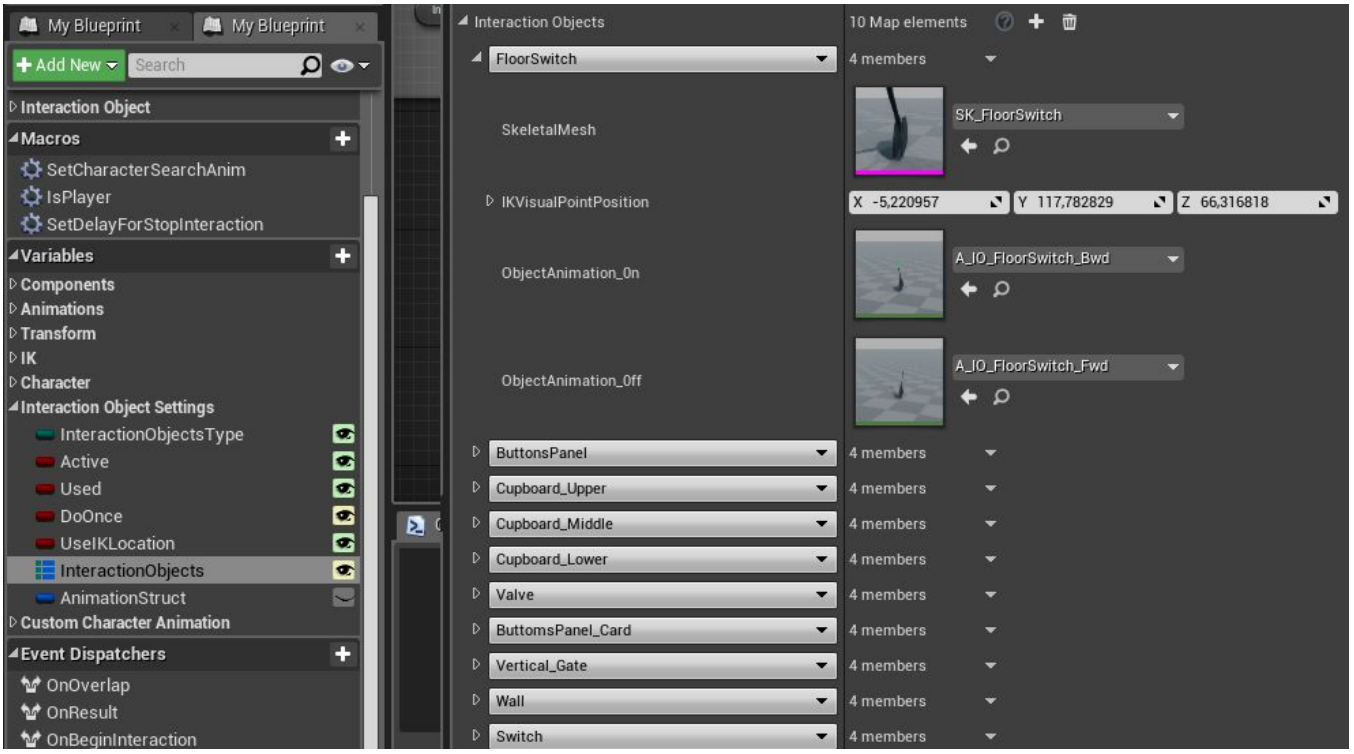
Объект необходимо добавить в enum (перечень объектов по которому выбираются настройки определенного объекта) Enum_InteractionObjects



Назовем наш новый объект “FloorSwitch” (названия объекта может быть произвольным).

Настройка нового объекта:

Следующим шагом надо настроить наш новый объект для этого открываем в BP_InteractionObject контент браузере. Добавляем в массив **Interaction Objects Array** новый объект посредством нажатия на + рядом кол-вом элементов, появится элемент New выбирете за место New ваш новый интерактивный объект:



Разберем все настройки по пунктам:

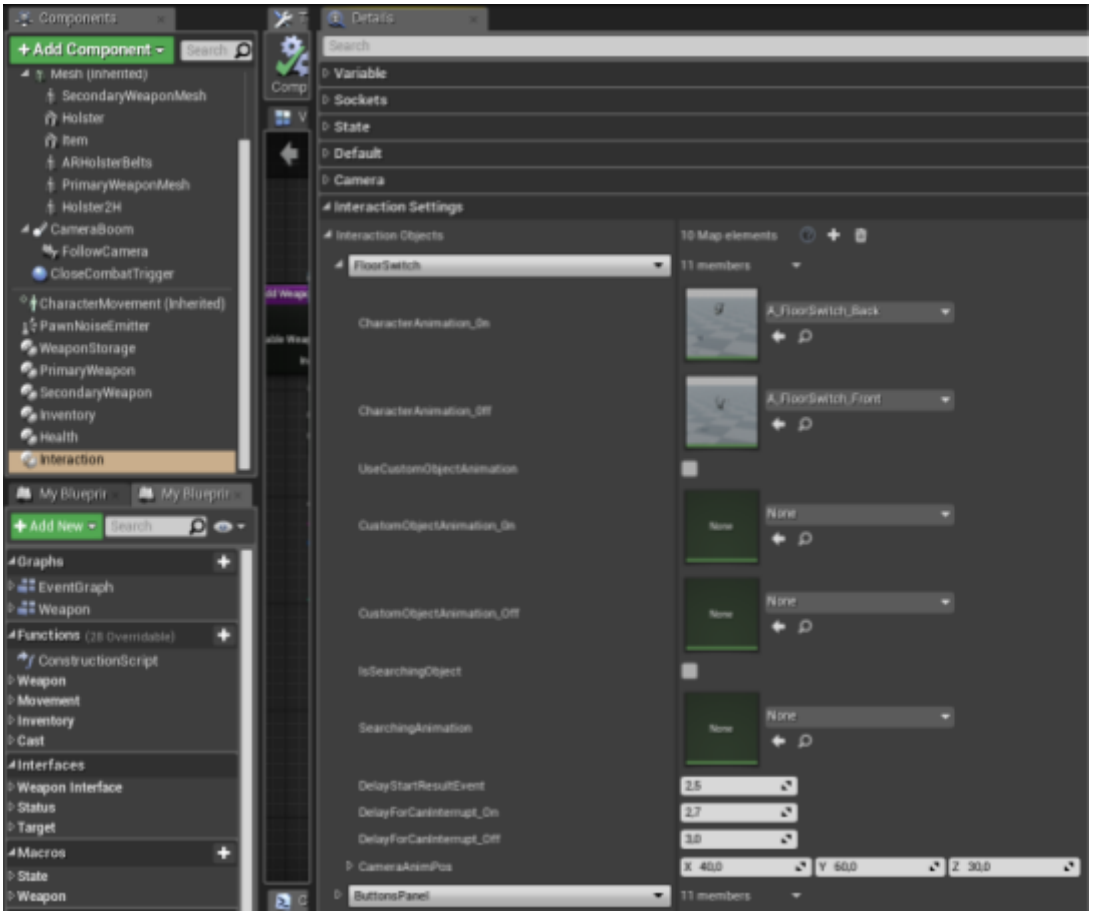
Skeletal_Mesh - модель интерактивного объекта

IKVisualPointPosition - расположение иконки руки по умолчанию

ObjectAnimation_0n - анимация включения объекта

ObjectAnimation_0ff - анимация выключения объекта

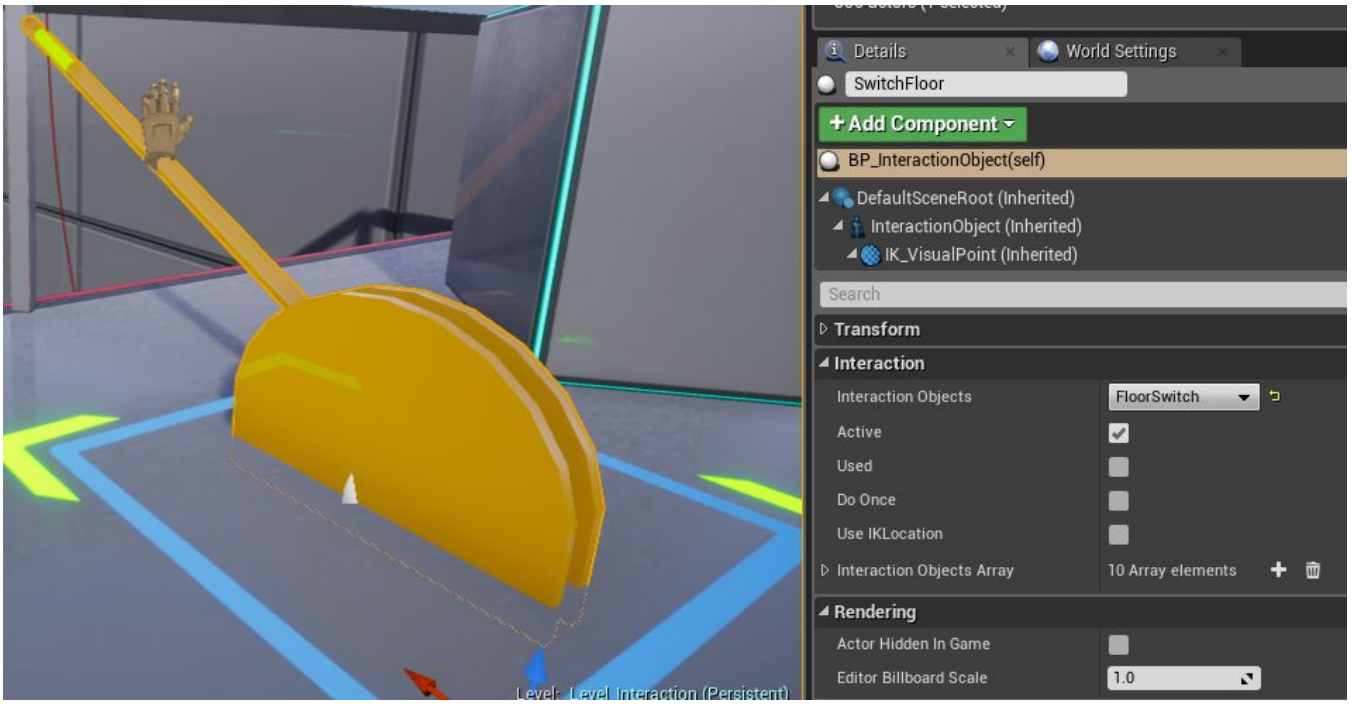
На этом наш новый объект готов. Теперь давай укажем настройки взаимодействия персонажа с данным объектом. Для этого перейдите в пешку игрока “Character_Pawn” (мы рассматриваем родительский класс, у вас может быть своя пешка в виде дочернего класса можете перейти в неё сразу) там выделите компонент Interaction и найдите справа настройки “Interaction Settings”. Далее раскройте список Interaction Objects и так же добавьте новый объект как до этого добавляли в IO классе.



Разберем все настройки по пунктам:
CharacterAnimation_0n - анимация включения у персонажа
CharacterAnimation_0ff - анимация выключения у персонажа
UseCustomObjectAnimation - использовать ли указанную ниже анимацию для объекта. Может пригодится когда к примеру ваша пешка не человек и неё свои анимации с объектом, либо человек но взаимодействует по другому с данным объектом.
CustomObjectAnimation_0n - новая анимация включения объекта. (используется только при включенном параметре *UseCustomObjectAnimation)*
CustomObjectAnimation_0ff - новая анимация выключения объекта. (используется только при включенном параметре *UseCustomObjectAnimation)*
IsSearchingObject - является ли объект обыскиваемым (не требуется для рычага)
SearchingAnimation - анимация в которой находится персонаж после включения объекта (используется при обыске) в данном случае не требуется
DelayStartResultEvent - задержка после которой сработает вызов диспатчера (который в свою очередь запускает ивент результата, другими словами к примеру выдвигание моста) через 2.5 секунды после начала анимации.
DelayForCanInterrupt_On - задержка после которой персонаж может прервать взаимодействие с объектом при включении.
DelayForCanInterrupt_Off - задержка после которой персонаж может прервать взаимодействие с объектом при выключении.
IKVisualPointPosition - расположение иконки руки по умолчанию
CameraAnimVector - положение камеры у персонажа при взаимодействии
Теперь новый объект добавлен и добавлены все настройки взаимодействия с ним.

Применения объекта на сцене:

Первым делом расположите **BP_InteractionObject** на сцену посредством перетаскивания из контент браузера на сцену. В деталях актора, во вкладке interaction выберите наш новый объект “FloorSwitch”.



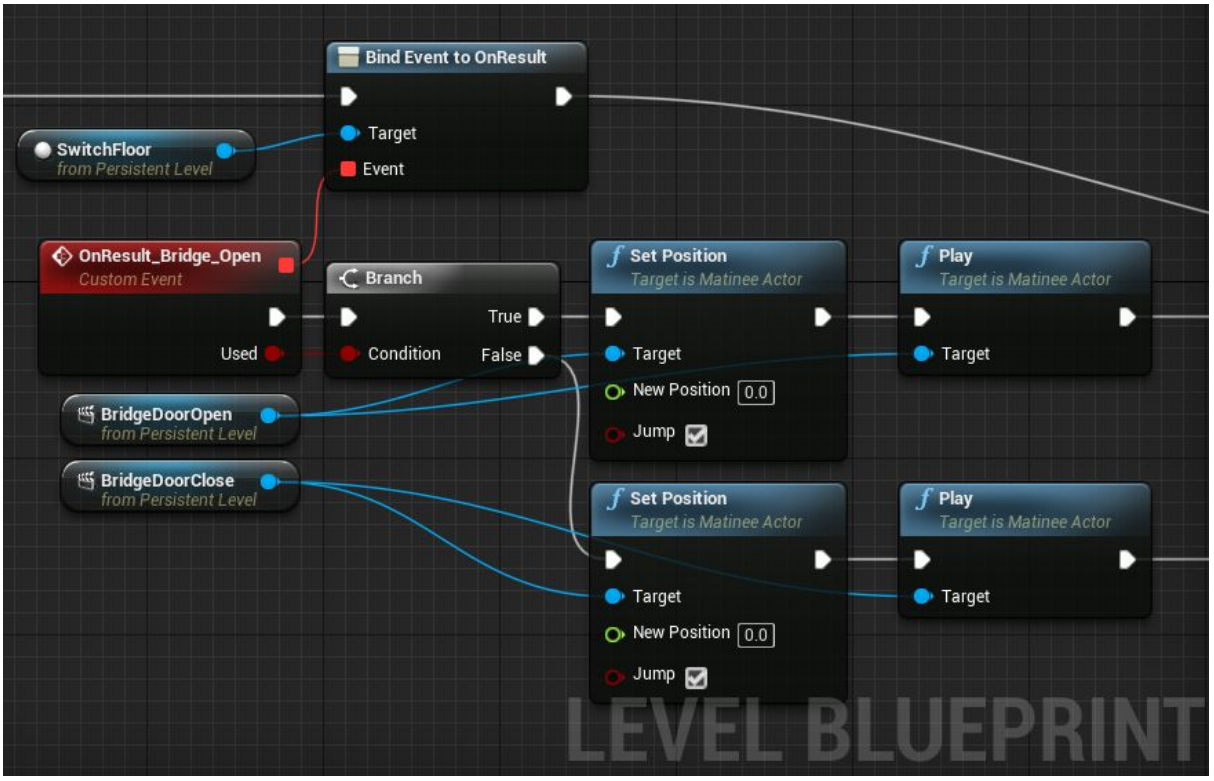
***Параметры Interaction**:*
Active - активный ли объект, Может ли персонаж использовать его в данный момент. Может изменяться динамически например для активации объекта нужно какое-то действие (поиск пароля или карты-ключа на примере кнопок на демо уровне.)
Used - включен ли объект изначально? Переключает on/off использования объекта
Do once - использовать объект единожды.
Use IKLocation - использовать IK положения рук на объекте.
Interaction Objects Array - содержит в себе массив настроек по умолчанию которые вы добавили для объектов, назначаются только текущему объекту на уровне.

Использование объекта на сцене:

Следующим шагом будет привязка анимации моста (или любого вашего действия) к взаимодействию рычага.

Выделяем объект на сцене и переходим в level blueprint и посредством правой кнопки мыши добавляем ссылку на наш объект на сцене.

Дальше привязываем нужный нам ивент к диспатчеру (в данном случае анимация моста)



На этом все - можно управлять мостом с помощью рычага.

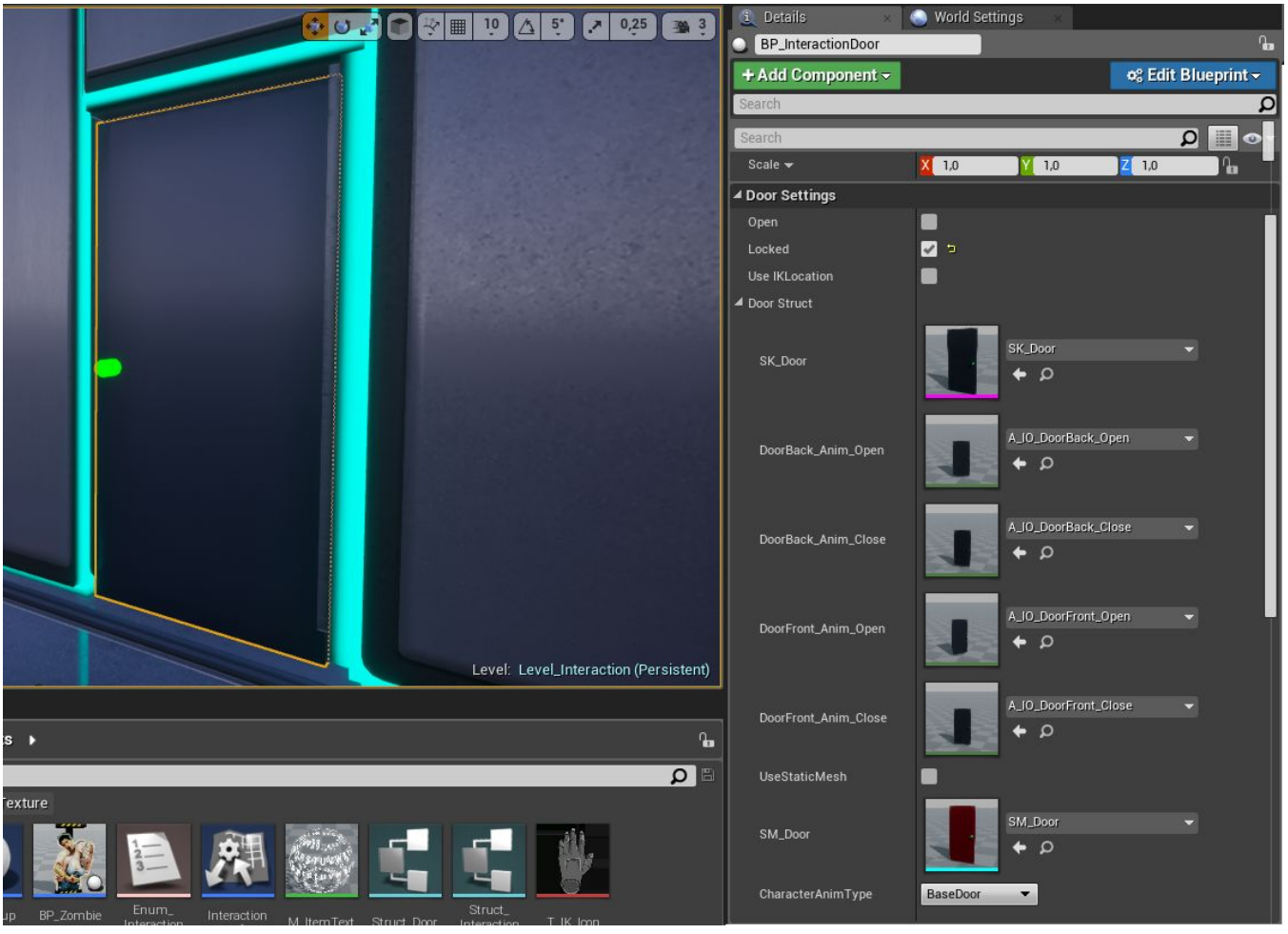
4. Двери :

Класс дверей предназначен для быстрого и удобного размещения дверей на вашем уровне. Дверь автоматически определяют с какой стороны персонаж и подключает необходимую логику.

Как это работает:
Система взаимодействия с дверьми работает по следующей схеме - в акторе “**BP_InteractionDoor**” присутствуют триггеры при пересечении которых персонаж может начать взаимодействие с дверью посредством нажатия кнопки взаимодействия (по умолчанию “E”). После нажатия кнопки происходит просчет в пешке - может ли в данный момент персонаж взаимодействовать с дверью (не стреляет ли он? не взаимодействует ли он уже с объектом в данный момент? и так далее) в случае если персонаж может взаимодействовать с дверью то BP_InteractionDoor запускается логика, которая подбирает необходимые анимации, плавно меняет расположение персонажа, учитывает с какой стороны двери находится персонаж… и впоследствии персонаж открывает дверь (если она не заперта).

Как разместить и настроить дверь:

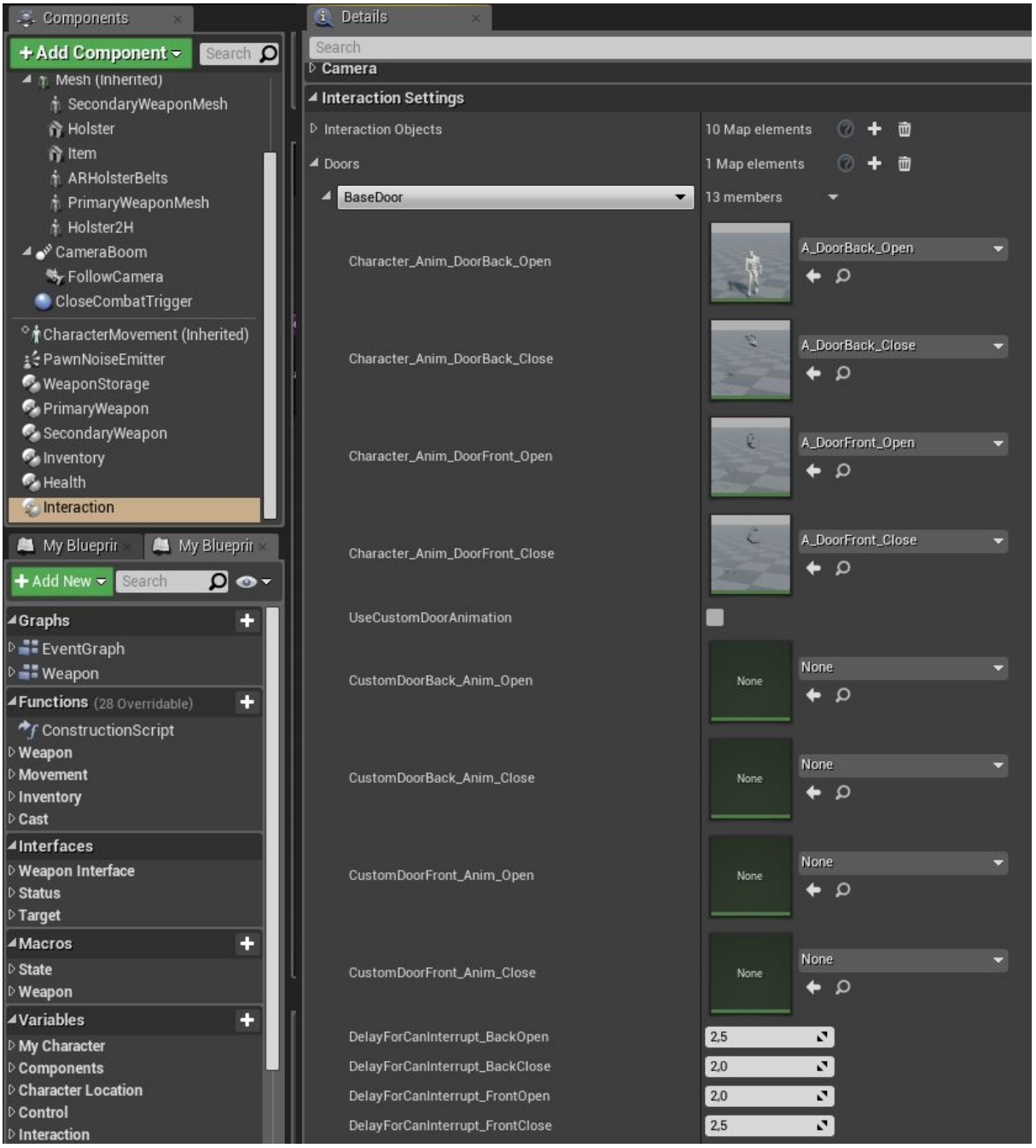
Первым делом расположите **BP_InteractionDoor** на сцену посредством перетаскивания из контент браузера на сцену. В деталях актора, во вкладке Door Setting рассмотрим параметры двери.



***Параметры Door Setting**:*
Open - открыта ли изначально дверь.
Locked - закрыта ли дверь (при взаимодействии вместо открывания двери будет вызван диспатчер *OnDoorLocked*. По данному диспатчеру вы можете придумать логику открывания двери на демо уровне это поиск ключа)
Use IKLocation - использовать IK положения рук на объекте.
Interaction Objects Array - содержит в себе массив настроек по умолчанию которые вы добавили для объектов, назначаются только текущему объекту на уровне.

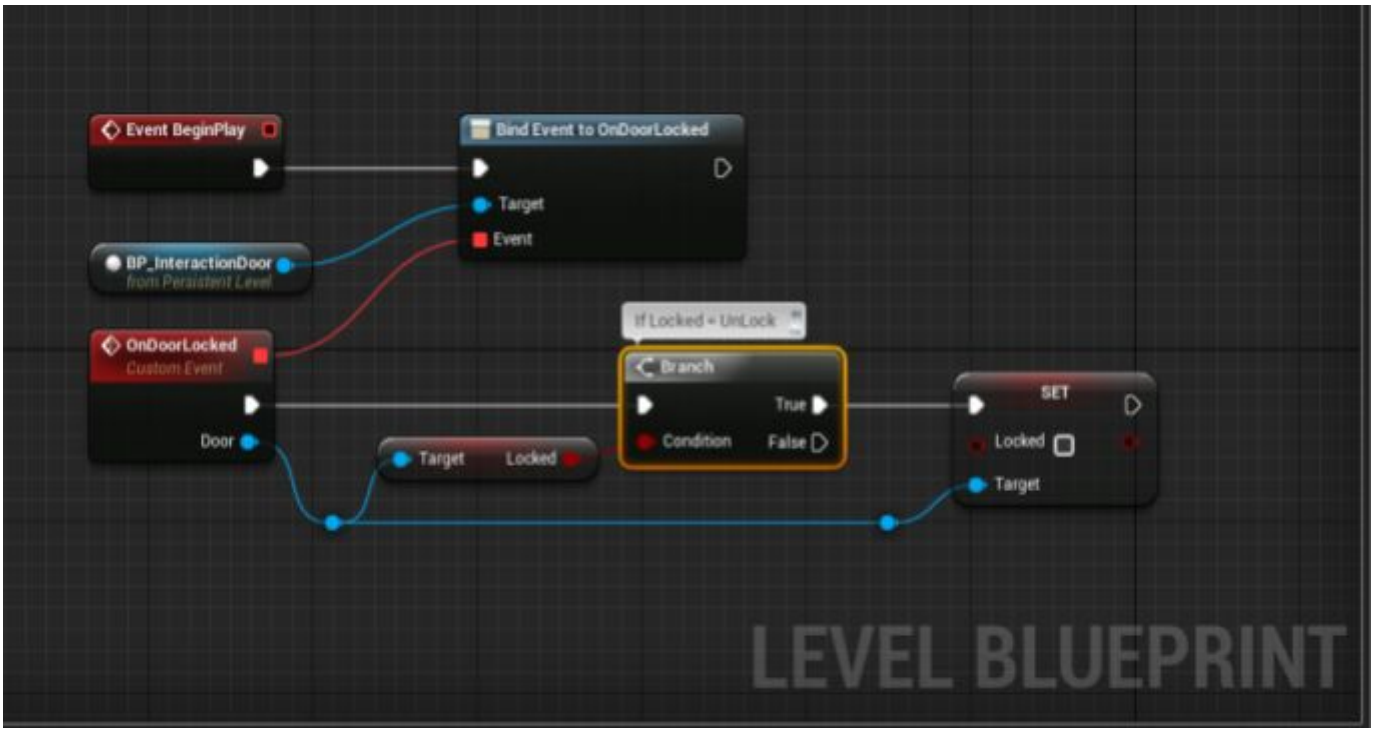
Структура двери - DoorStruct:
SK_Door - Выбор скелетал меша двери, если вы хотите использовать скелетную дверь то вам необходимо заскинить вашу дверь на текущий скелет двери, чтобы использовать анимации по умолчанию.
Door_anim... - Эти параметры указывают анимацию двери. Вы можете их менять в любой момент - на примере уровня *ZombieShootingRange* там дверь выбивается и использует другую анимацию.
UseStaticMesh - Этот параметр отвечает за использование статичного меша.
SM_Door - Выбор статичного меша для двери.
CharacterAnimType - Тип взаимодействия для персонажа. Вы можете настроить несколько типов взаимодействия, к примеру вышибить дверь, или медленно её открыть и т.д. Вы также можете не создавать несколько типов, (если к примеру нужно определенное взаимодействие только с этой дверью) а просто указать анимацию персонажа при помощи блока настроек “Custom Character Animation”. Настройки находятся так же в классе дверей.

Настройки у персонажа взаимодействия с дверью:
Перейдите в пешку игрока “Character_Pawn” (мы рассматриваем родительский класс, у вас может быть своя пешка в виде дочернего класса можете перейти в неё сразу) там выделите компонент Interaction и найдите справа настройки “Interaction Settings”. Далее раскройте список Doors там будут типы взаимодействия, по умолчанию оно одно “BaseDoor”.



Door_anim... - Эти параметры указывают анимацию персонажа.
UseCustomDoorAnimation - использовать ли указанную ниже анимацию для данного типа дверей. Может пригодится когда к примеру ваша пешка не человек и неё свои анимации с дверью, либо человек но взаимодействует по другому с данной дверью.
DelayForCanInterrupt.. - задержки по времени через которые можно прервать анимацию персонажа.

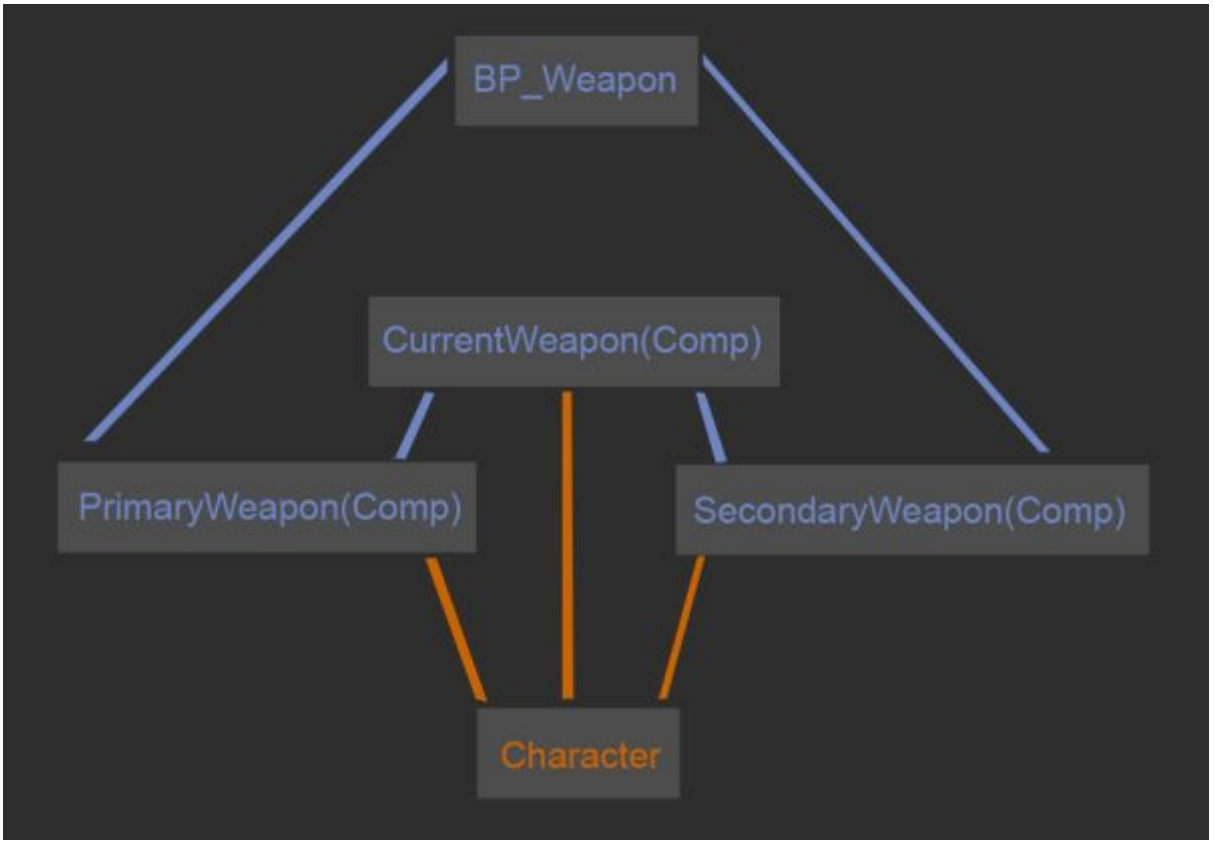
Все новая дверь готова к использованию. Дверь вызывает два диспатчера на которые вы можете придумать дополнительную логику в момент открывания двери *OnDoorOpen* и если дверь заперта *OnDoorLocked.* Обращаться к данным диспатчерам вы можете так же как в рубрике ***Использование объекта на сцене.***



Огнестрельное Оружие

Класс оружия в character interaction позволяет создавать огнестрельное оружие различных типов. Класс оружия представлен в виде актор компонента. По умолчанию персонаж имеет 2 компонента оружия (первичное и вторичное оружие).

Реализована следующая схема взаимодействия оружия и персонажа:

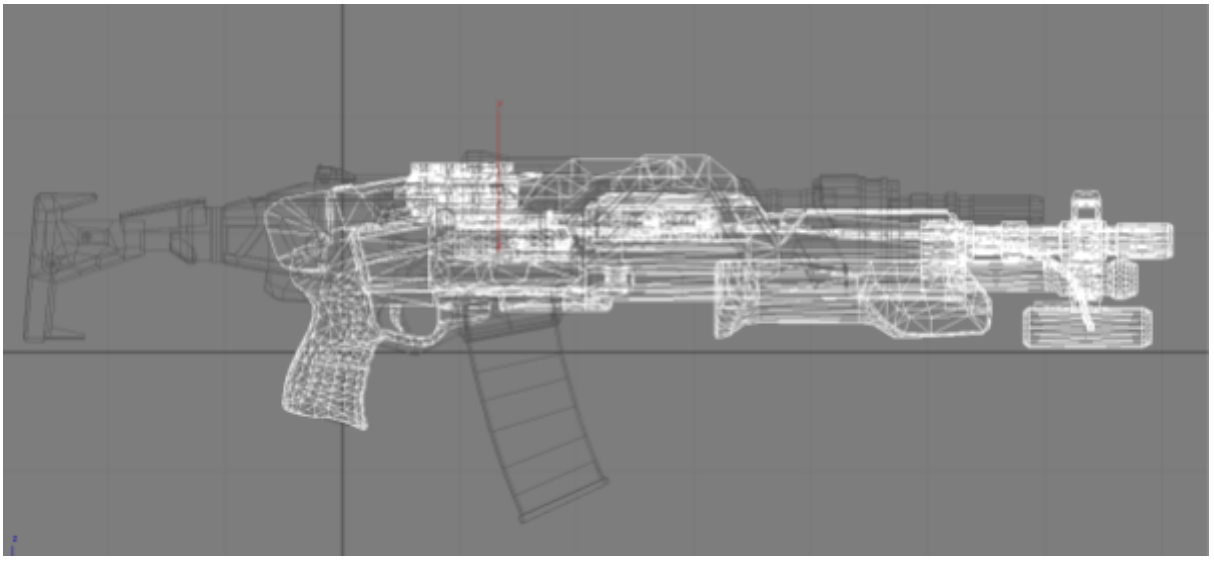


Другими словами персонаж имеет 3 компонента оружия “**BP_Weapon**”, первичное оружие, вторичное оружие и текущее оружие.

Сам класс оружия называется “**BP_Weapon**” данный класс содержит в себе всю логику огнестрельного оружия, но для более удобного создания и хранения оружия был создан класс “**WeaponStorage**”, данный класс работает со структурами компонентов оружия и хранит их под индексами. Так он заменяет структуру вышеуказанных компонентов для использования оружия. Так для переключения между оружиями или их смены был создан виджет “**Hud_WeaponManager**” данный виджет образует связь между компоннетом “**WeaponStorage**” и пешкой(персонажем).

Добавление огнестрельного оружия

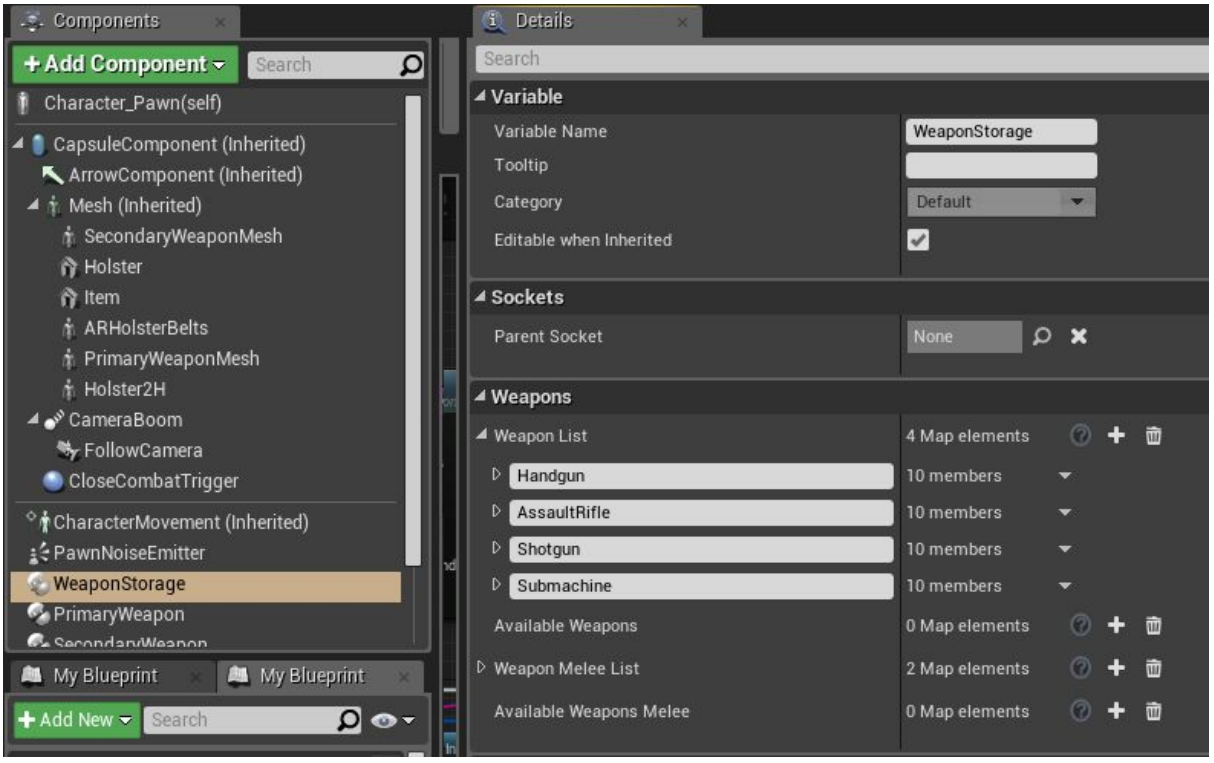
Подробно рассмотрим как добавляется новое оружие на примере дробовика. В первую очередь добавим весь необходимый контент для дробовика.
Во первых создайте модель дробовика таким образом чтобы начальные координаты были по центру рукоятки как показано на изображении - тем самым вам не придется его дополнительно выравнивать (но вы сможете выравнивать его в настройках оружия). Создайте скелет для оружия если это необходимо (затворы, помповая рукоятка и прочее). Если ваше оружие не анимируется то при импорте укажите укажите что оружие скелетал меш. Оружие в Character Interaction должно быть скелетал меш.



Дробовик присутствует в проекте (обновление 3)

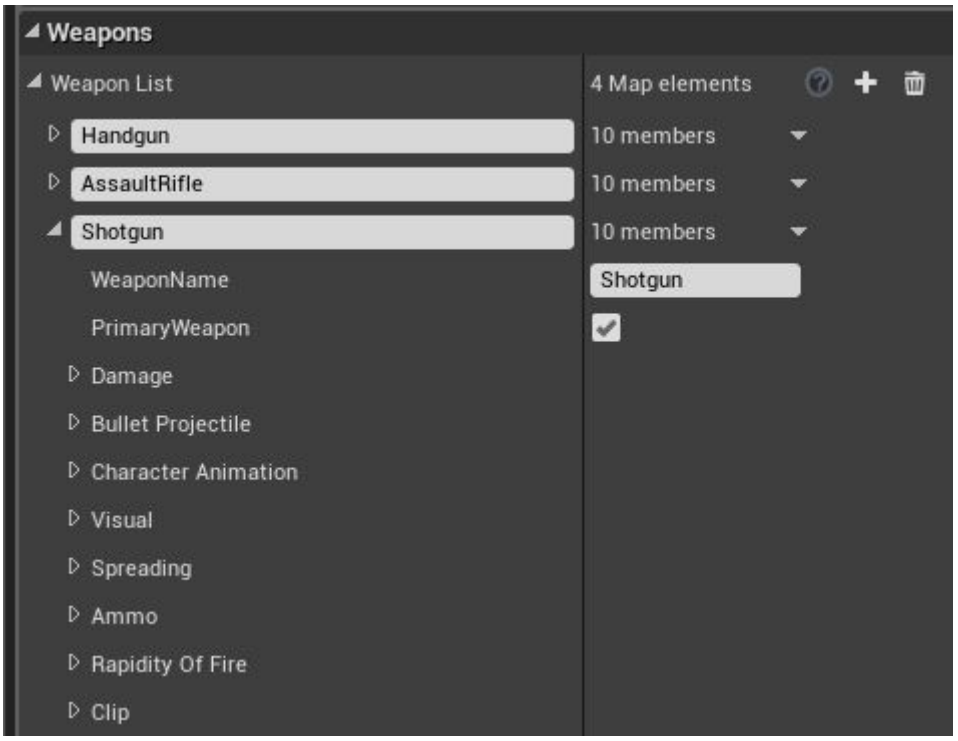
После создания модели оружия переходим к добавлению необходимых сокетов:
Socket_Muzzle - место проигрывания вспышки от оружия
Socket_Shell - место вылета гильз

Socket_Clip - место вылета обоймы (на примере дробовика это не нужно т.к. обойма не предусмотрена)
Также если необходимо добавьте в проект звуки выстрела, перезарядки.
После добавления контента в проект отправляемся на склад оружия в компонент “**WeaponStorage**”, чтобы настроить и добавить его в игру. Для этого откройте пешку “**Character_Pawn**” и выделите компонент **WeaponStorage,** справа вы увидите его настройки.



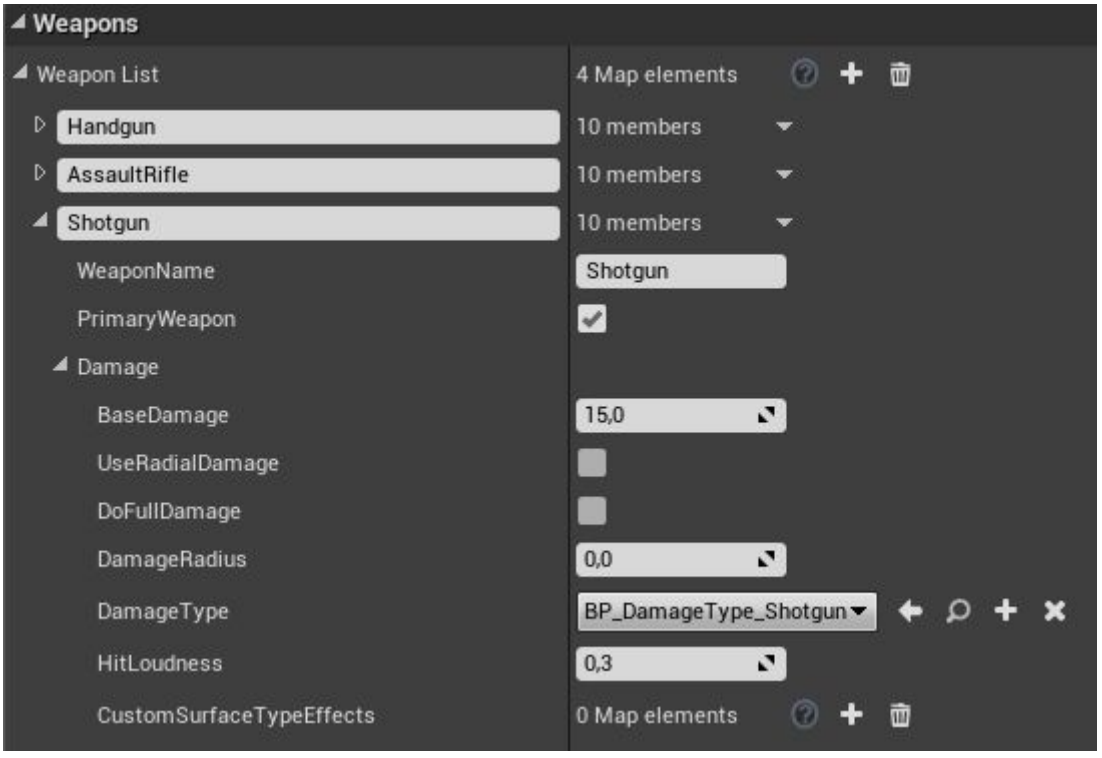
Weapon list - это список всего оружия в проекте, добавляйте сюда все оружие которое может использовать данный персонаж, для каждого персонажа вы можете настраивать свой список, если оружие должны использовать все персонажи я рекомендую добавлять оружие в родительский класс “**Character_Pawn**” и в дальнейшем вс едочерние классы уже будут знать о добавленом оружии.

Добавьте наше новое оружие дробовик посредством кнопки + напротив WeaponList. Назовите оружие - на нашем примере это “**Shotgun**”. Начнем по порядку настраивать наше оружие.



WeaponName - Техническое название оружия, называйте так же как вы назвали новый индекс в массиве, на данном примере - Shotgun
PrimaryWeapon - Является ли оружие первичным? Данный параметр определяет к какому из двух типов отнести оружие. Либо первичное (двуручное) либо вторичное (одноручное). В нашем случае оружие первичное - ставим галку во включенное положение.

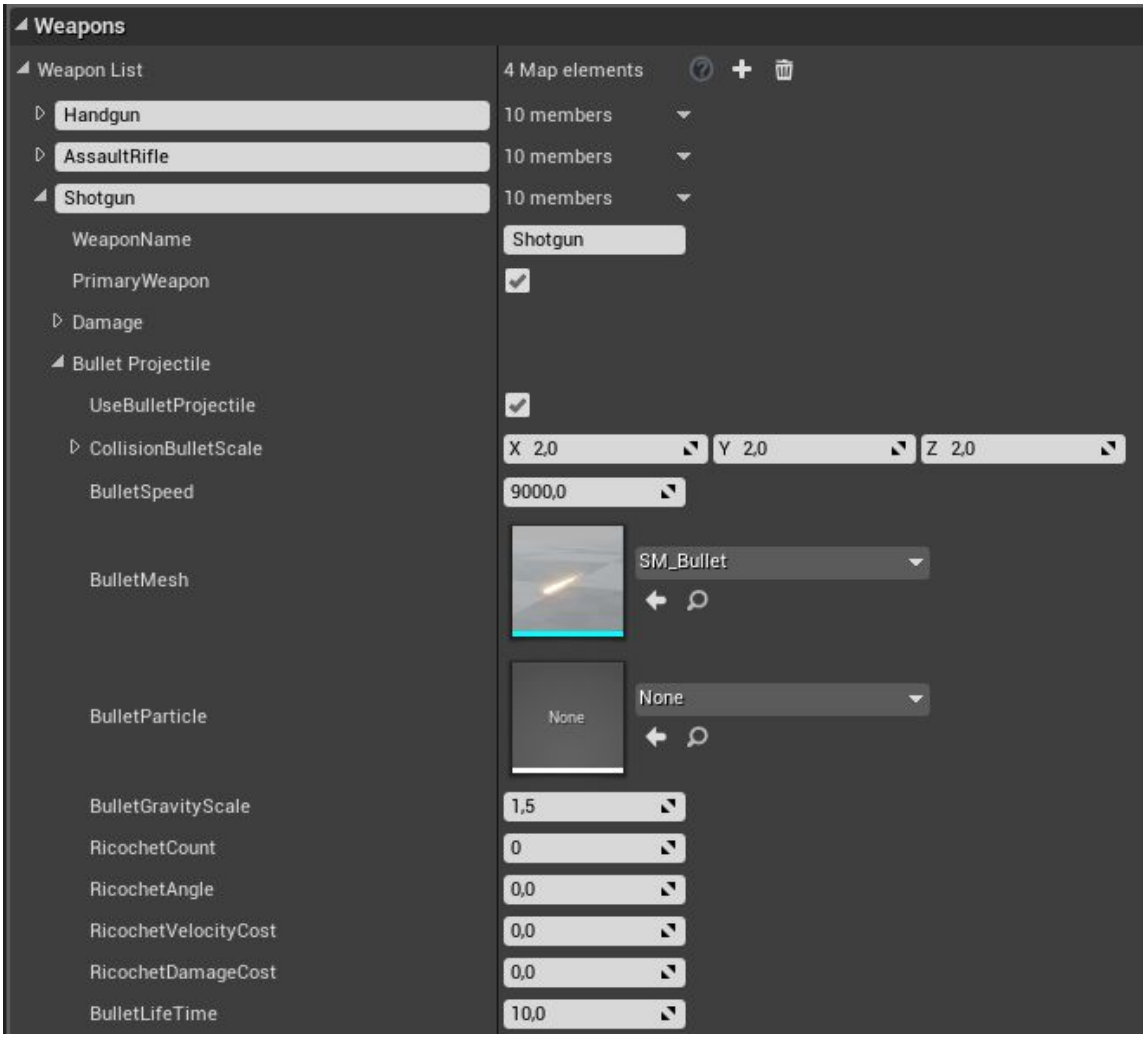
Настройки повреждения:



BaseDamage - Базовое повреждения от оружия при попадании. Так как у нас дробовик и будет лететь несколько снарядом можно указать повреждения небольшое.
UseRadialDamage - Будут ли снаряды наносить повреждение по площади. Например взрыв от ракеты.

DoFullDamage - относится к повреждению по площади. При включенном парамтере будет наносится повреждение полностью. При выключенном повреждение будет снижаться в зависимости от расстояния попадания снаряда.
DamageRadius - относится к повреждению по площади. Радиус рассчитывается в сантиметрах.
DamageType - Выбор класса типа повреждения. В данном классе настраиватся импульс и повреждения для разрушаемых объектов. Более подробно здесь
HitLoudness - Громкость попадания снаряда. Не влияет на громкость звука, это технический парамтер для ИИ
CustomSurfaceTypeEffects - тут указываются дополнительные эффекты попадания или заменяются для данного оружия. К примеру елси это ракета то должен быть взрыв, декаль воронки и соответствующий звук. тут вы можете настроить эффект на любую поверхность присутствующую в проекте (SurfaceType).

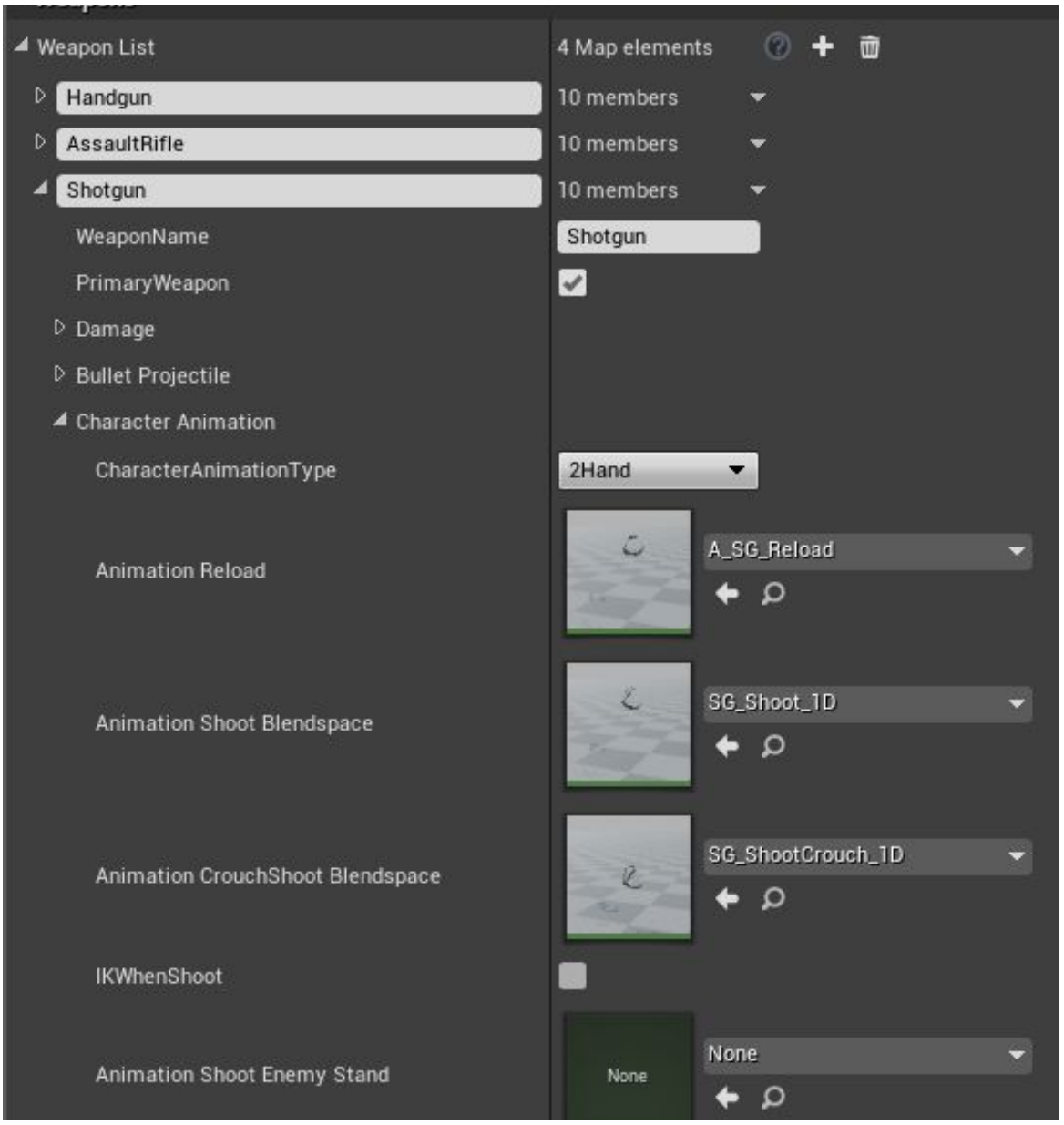
Настройки снаряда:



UseBulletProjectile - тип снаряда. При включенном параметре снаряд будет иметь визуальное представление и лететь по заданной траектории, при выключенном параметре снаряд будет моментально достигать цели.
CollisionBulletScale - Множитель коллизии снаряда (размер снаряда)
BulletSpeed - Скорость полета снаряда
BulletMesh - Модель снаряда
BulletGravityScale - Множитель силы гравитации на снаряд. Чем больше тем быстрее снаряд упадет.
RicochetCount - Кол-во раз которое снаряд может отрикошетить от поверхности.

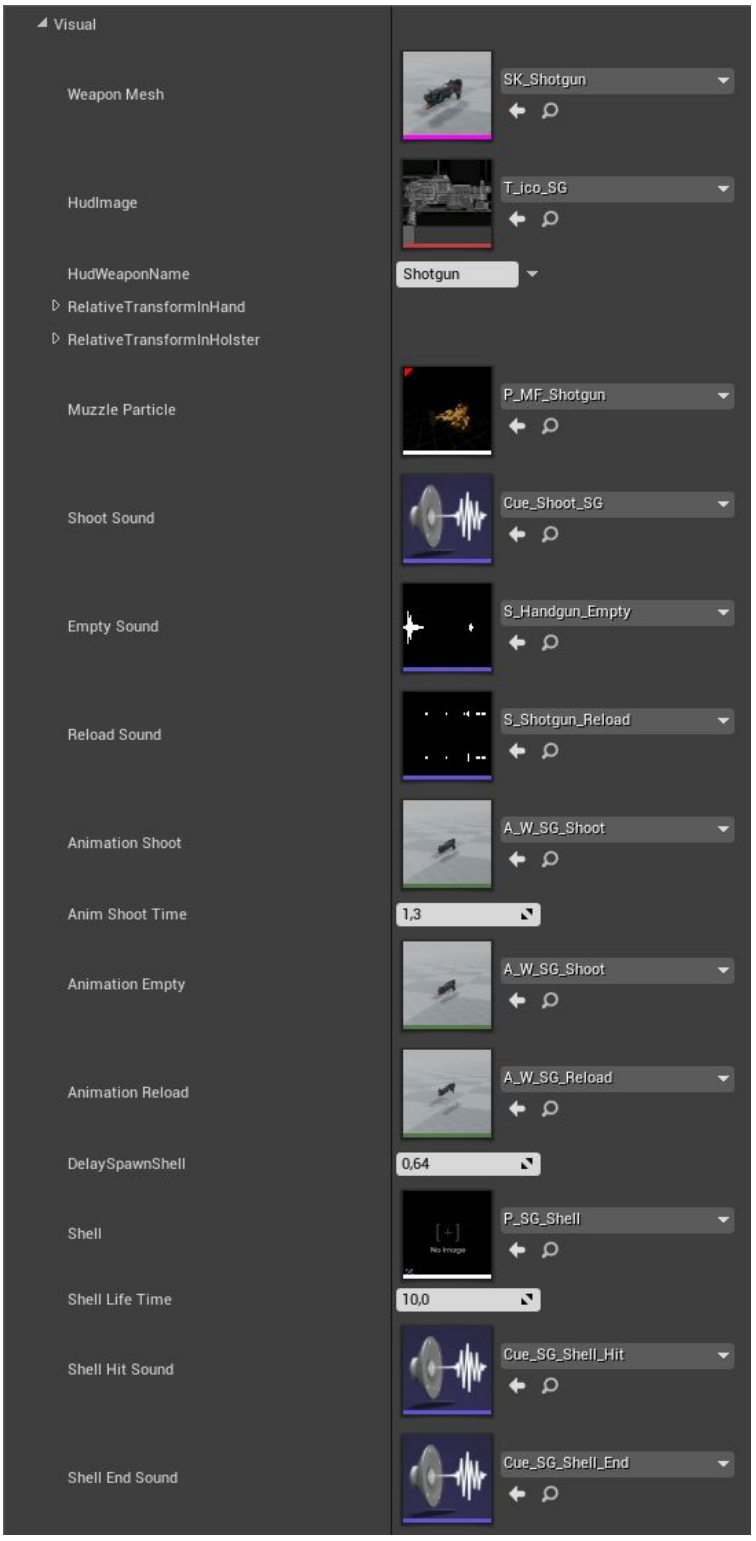
RicochetAngle - Угол при котором возможен рикошет
RicochetVelocityCost - Сколько процентов скорости снаряд теряет при рикошете указывается от 0-1
RicochetDamageCost - Сколько процентов повреждения снаряд теряет при рикошете указывается от 0-1
BulletLifeTime - Максимальное время в секундах при котором снаряд будет жить. (оптимизация)

Настройки анимации персонажа с оружием:



CharacterAnimationType - Тип анимации персонажа с данным оружием: 2H - персонаж использует это оружие как двуручное, 1H - персонаж использует это оружие как одноручное.
AnimationReload - Анимация перезарядки персонажа с данным оружием
AnimationShootBlendpace - Анимация стрельбы персонажа с данным оружием
IKWhenShoot - использование ик для левой руки при стрельбе. Так как у нас дробовик и подвижная помпа то выключаем параметр.
AnimationShootEnemyStand - Анимация стрельбы стоя для ИИ

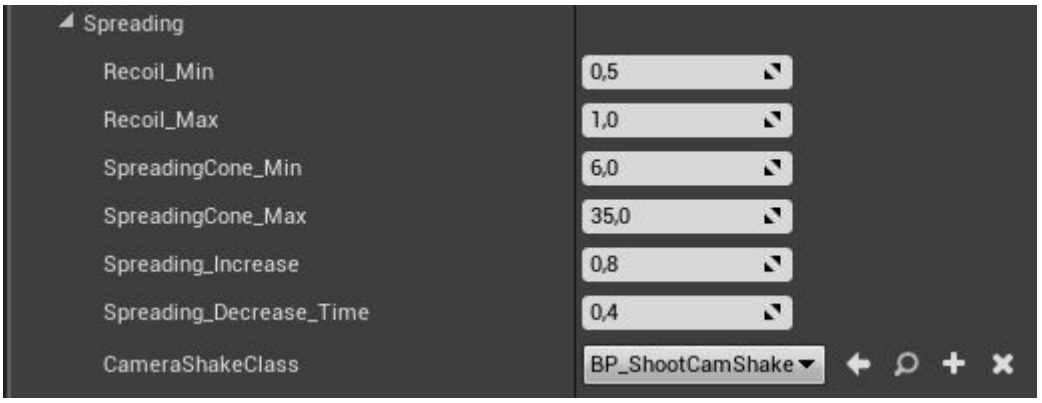
Визуальные настройки оружия:



WeaponMesh - Модель оружия
HudImage - Иконка оружия для HUD
HudWeaponName - имя оружия для HUD
RelativeTransformHand - Относительные координаты расположения оружия от руки персонажа.
RelativeTransformHolster - Относительные координаты расположения оружия от кобуры.
Muzzle Particle - Эффект от выстрела.
Shoot Sound - Звук выстрела оружия.
Empty Sound - Звук щелчка курка при отсутствии патронов
Reload Sound - Звук перезарядки оружия
Animation Shoot - анимация выстрела оружия
Animation Empty - анимация выстрела оружия при последнем патроне

Animation Reload - анимация перезарядки оружия
DelaySpawnShell - задержка выпадата гильзы. Так как гильза у дробовика должна вылетать не сразу то указываем задержку
Shell - эффект гильзы
Shell Life Time - время жизни гильзы
ShellHitSound - Звук от удара гильзы
ShellEndSound - Звук последнего удара гильзы

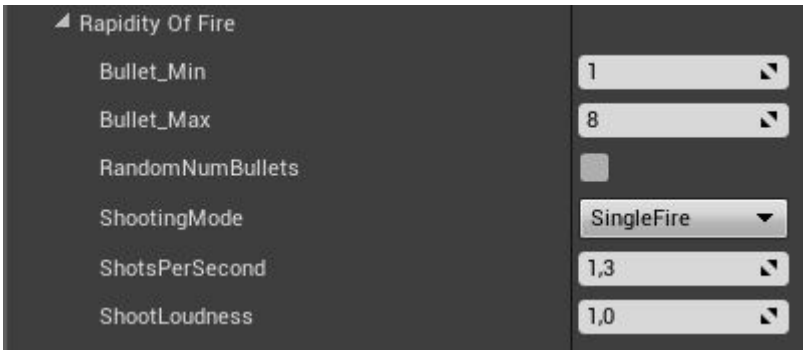
Настройки разброса оружия:



Recoil_Min - минимальный (изначальный при первом выстреле) множитель отдачи (тряска камеры)
Recoil_Max - максимальный множитель отдачи (тряска камеры)Чем больше значение тем больше отдача оружия при непрерывной стрельбе.
SpreadingCone_Min - минимальный (изначальный при первом выстреле) конус разброса оружия. Отвечает за максимальную точность оружия. Чем меньше значение тем точнее оружие.
SpreadingCone_Max - максимальный конус разброса оружия. Отвечает за максимальную разброс оружия. Чем больше значение тем больше разброс оружия при непрерывной стрельбе.
Spreading_Increase - отвечает насколько увеличивается разброс при выстреле. Чем выше значение тем быстрее оружие достигает максимального разброса.
Spreading_Decrease_Time - за сколько времени в секундах стрельба восстанавливается в точности.
CameraShakeClass - класс тряски камеры, можете использовать по умолчанию увеличивая или уменьшая тряску параметрами Recoil_Min и Recoil_Max.
Настройки амуниции:

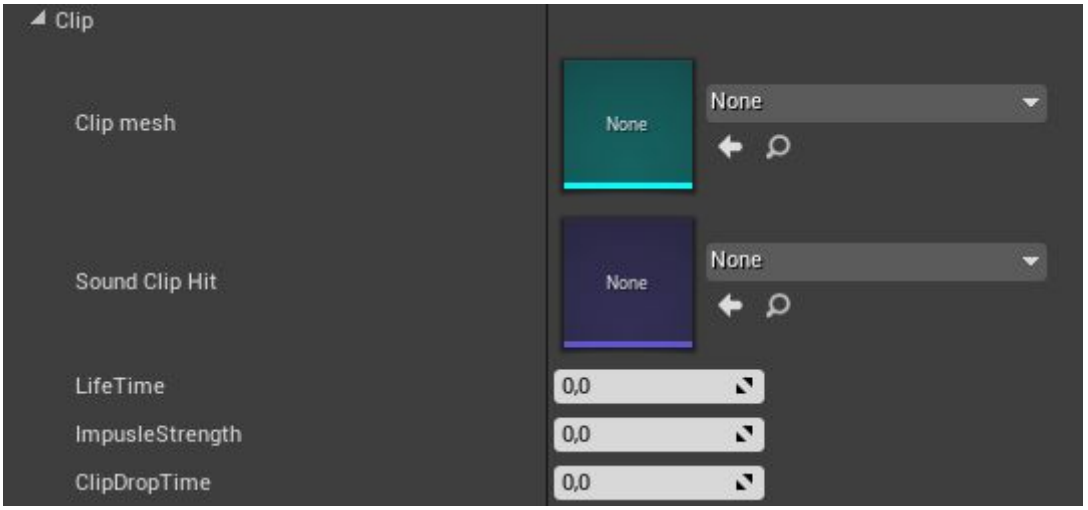
AmmoType - тип патрон, более подробно ниже в описании инвентаря. Вкратце какой тип патрон использует данное оружие.
Ammo_ClipSize - Максимальный размер обоймы оружия
Ammo_InClip - Изначальное кол-во патрон в обойме
ReloadTime - Время перезарядки (указывается от анимации)

Настройки скорострельности:



Bullet_Min - Минимальное количество снарядов при выстреле
Bullet_Max - Максимальное количество снарядов при выстреле
RandomNumBullets - Рандомное количество снарядов от значения Bullet_Min до значения Bullet_Max
ShootingMode - указывает тип стрельбы оружия. *Single Fire* - одиночная стрельба. *AutoFire* - автоматическая стрельба.
ShotsPerSecond - максимальное кол-во выстрелов в секунду.
ShootLoudness - Громкость выстрела. Не влияет на громкость звука, это технический параметр для ИИ. В виде примера добавлено оружие с глушителем где громкость от выстрела минимальна и враг не реагирует.

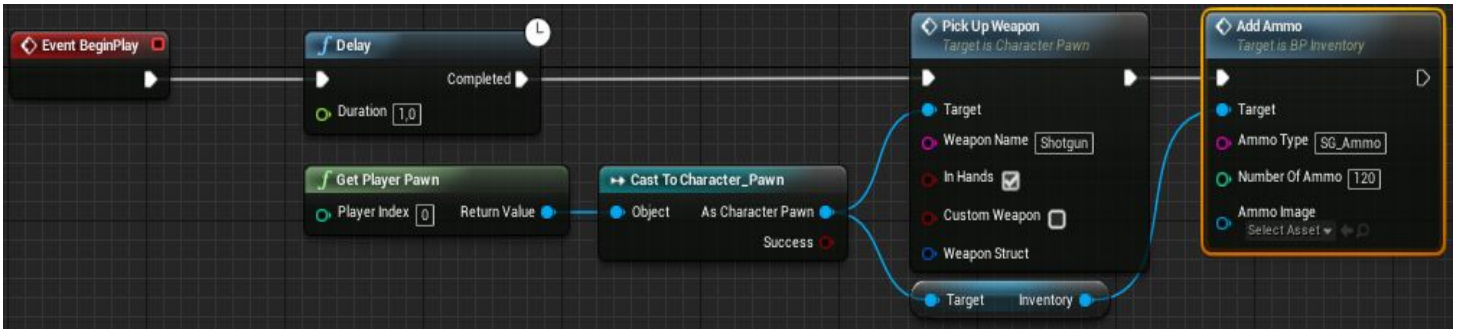
Настройки вылетающей обоймы:



ClipMesh - Модель обоймы
Sound Clip Hit - Звук от удара обоймы
LifeTime - Время жизни обоймы
ImpusleStrength - Сила импульса с которой вылетает обойма (направление указывается сокетом Socket_Clip в оружии)
ClipDropTime - задержка перед вылетом обоймы (указывается от анимации)

На этом все наше оружие добавлено в игру и имеет свои настройки. Чтобы протестировать оружие добавив его персонажу используйте следующие узлы к примеру в блупринте уровня. Внимание если вы добавляете оружие сразу как только начинается игра (BeginPlay) то задайте задержку для инициализации

системы.



Немного ниже рассмотрим как добавить оружие в виде предмета и как его переключать при помощи “**WeaponManager**”

Оружие ближнего боя

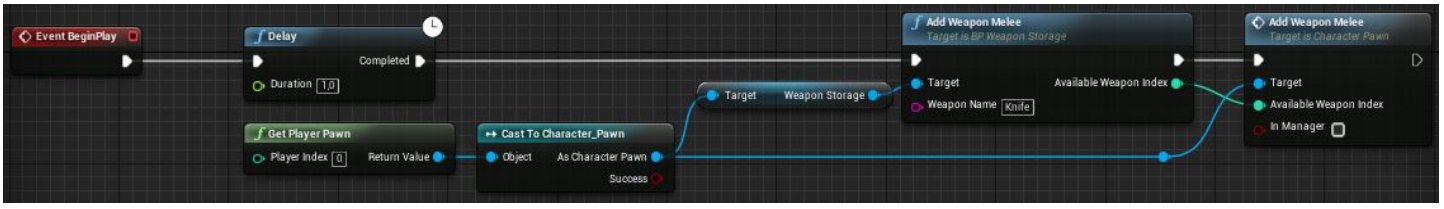
По умолчанию оружие ближнего боя используется персонажем, когда он не целится, что позволяет комбинировать бой с ближним и дальним оружием. Оружие ближнего боя работает по такому же принципу как и огнестрельное. Чтобы добавить оружия ближнего боя - откройте пешку “**Character_Pawn**” и выделите компонент **WeaponStorage,** справа вы увидите его настройки. Перейдите в список оружия ближнего боя “**Weapon Melee List**”. Оружие “**Unarmed**” является базовым оружием, которое добавляется по умолчанию и применяется если вы не вооружены оружием ближнего боя или снимаете с персонажа оружие. Добавление оружия ближнего боя осуществляется нажатием на + напротив WeaponMeleeList.

Назовите ваше новое оружие. Рассмотрим настройки оружия:


Weapon Melee List
2 Map elements
Unarmed
6 members
Knife
6 members
WeaponName
Knife
WeaponMeshForItem
SM_Knife
RelativeTransformForItem
HudImage
T_Ico_Melee_Knife
HudWeaponName
Knife
Attacks
1 Array elements
0
9 members
AttackType
Knife
1Hand
A_KN_Attack_01
2Hand
A_KN_Attack_01
Unarmed
A_KN_Attack_01
DamageStruct
1 Array elements
0
7 members
DelayForDealDamage
0,33
Damage
60,0
DamageAnim
Base
ImpusleStrength
800,0
ImpusleForCarMultiply
0,2
Distance
120,0
HitLoudness
1,0
DelayForInterrupt
1,1
HideWeapon
ItemInHand
ItemInHand
ItemMesh
SM_Knife
SocketName
Socket_Item_RHand
Transform
SoundAttack
A_Knife_Attack
Available Weapons Melee
0 Map elements

WeaponName - Техническое название оружия, называйте так же как вы назвали новый индекс в массиве, на данном примере - Knife
WeaponMeshForItem - Модель оружия для предмета, используется когда вы выбрасываете оружие на пол через WeaponManager
RelativeTransformForItem - Относительное положение для предмета.
HudImage - Иконка для оружия
HudWeaponName - название оружия для HUD
Attacks - лист атак для данного оружия. Расширяется посредством нажатия на + напротив Attacks
1Hand - Анимация удара когда в руках одноручное оружие
2Hand - Анимация удара когда в руках двуручное оружие
Unarmed - Анимация удара когда в руках нету оружия
*DamageStruc*t - Структура содержит в себе параметры повреждения, можно назначить на один удар несколько повреждений например если это комбо удар
DelayForDealDamage - Задержка в секундах перед нанесением повреждения, внимание если у вас несколько повреждений то следующая задержа будет считаться относительно предыдущей.
Damage - Сила урона
DamageAnim - название анимации повреждения у врага, например персонаж может упасть, либо отлетать анимационно в определенную сторону то настройте эту анимацию у врага и дайте ей название. У каждого врага может быть своя реация на этот удар
ImpulseStrength - Сила импульса удара по физическим объектам
ImpulseForCarMultiply - Сколько процентов от базового импульса наносить машине от 0-1
Distance - Дистанция в см при которой будет нанесено повреждение
HitLoudness - Громкость удара для ИИ
DelayForInterrupt - Через какое время можно будет прервать атаку
HideWeapon - Скрывать огнестрельное оружие при атаке
ItemInHand - Структура параметров для отображения предмета в руке.
ItemMesh - Модель оружия
SocketName - сокет к которому будет прикреплено оружие, по умолчанию добавлено 2 сокета для левой и правой руки:Socket_Item_LHand,Socket_Item_RHand.
Transform - Локация оружие относительно сокета
SoundAttack - звук атаки

На этом все наше оружие добавлено в игру и имеет свои настройки. Чтобы протестировать оружие добавив его персонажу используйте следующие узлы к примеру в блупринте уровня. Внимание если вы добавляете оружие сразу как только начинается игра (BeginPlay) то задайте задержку для инициализации системы.



Инвентарь и подбираемые предметы

Инвентарь служит для хранения различных предметов, которые подбирает персонаж. Инвентарь представлен в виде компонента “**BP_Inventory**” и содержит в себе логику различных примеров. По умолчанию инвентарь поддерживает следующие типы предметов:

Weapon - Огнестрельное оружие. Подбирая предметы такого типа инвентарь запускает логику добавления оружия в “**WeaponStorage**” у пешки.

WeaponMelee - Оружие ближнего боя. Подбирая предметы такого типа инвентарь запускает логику добавления оружия в “**WeaponStorage**” у пешки.

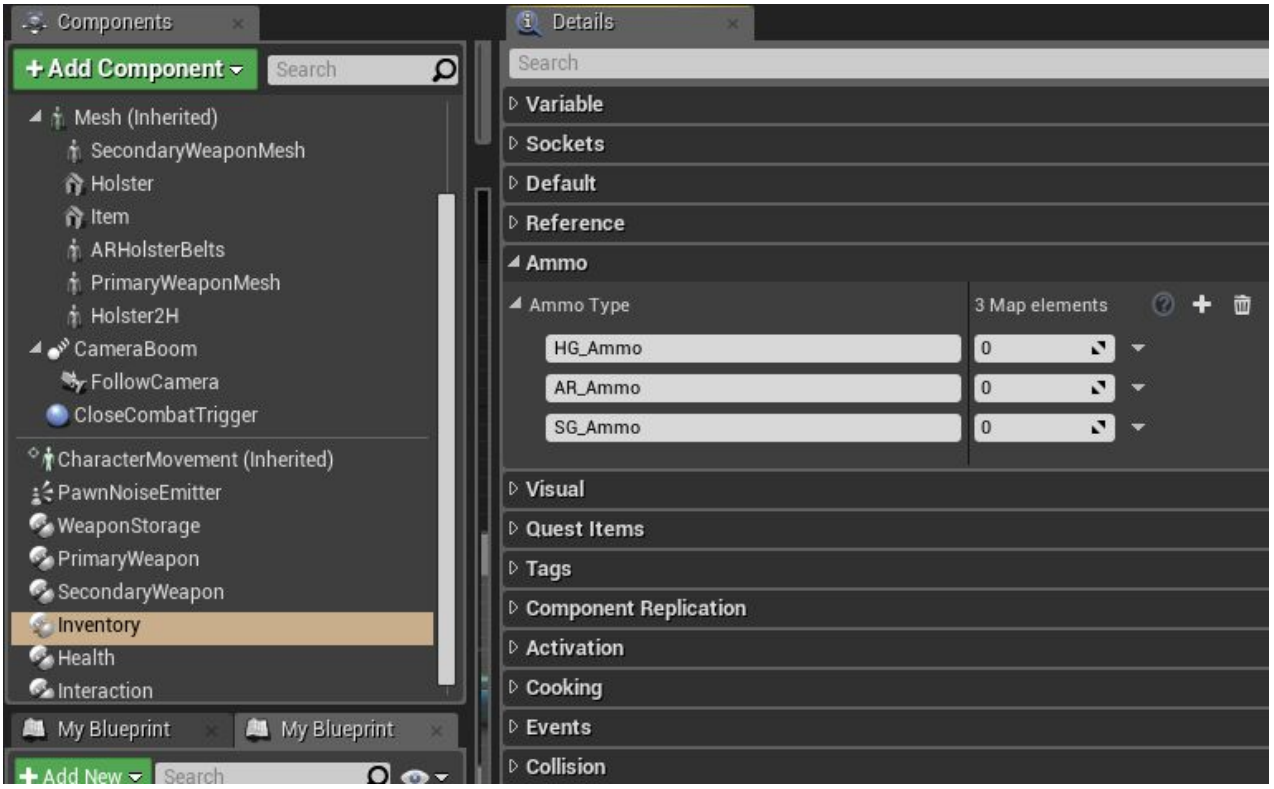
Ammo - Патроны. Огнестрельное оружие использует боеприпасы напрямую из инвентаря. Каждому оружию нужно указать какой тип боеприпасов оно использует. По умолчанию в инвентаре используется 3 типа боеприпасов:

HG_Ammo - патроны для пистолета и пистолет автомата

AR_Ammo - Патроны для штурмовой винтовки

SG_Ammo - Патроны для дробовика

Вы можете добавлять любое кол-во типов патронов. Для этого выделите компонент “Inventory” в пешке игрока и перейдите в соответствующую рубрику настроек “Ammo”.



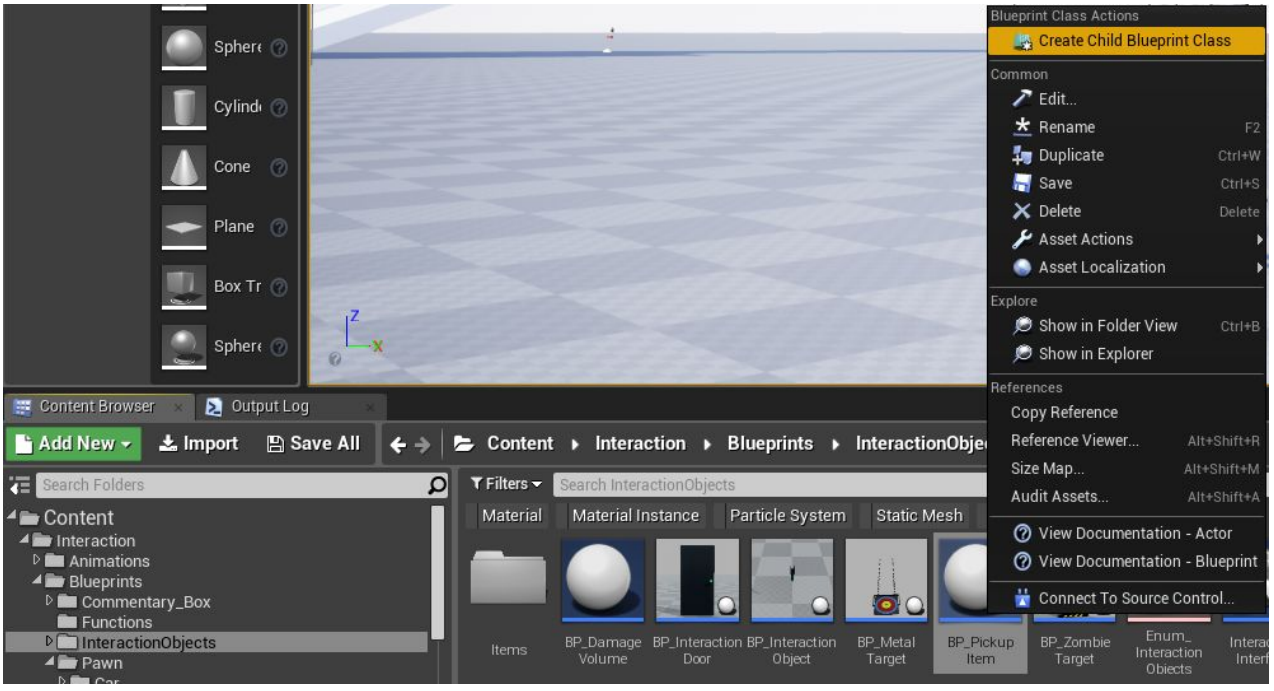
И последний тип предметов для инвентаря это:

QuestItem - данный предмет может быть чем угодно, как ресурсом для крафта чего либо, так и ключом от двери и т.д.

Рассмотрим все типы предметов на практике.

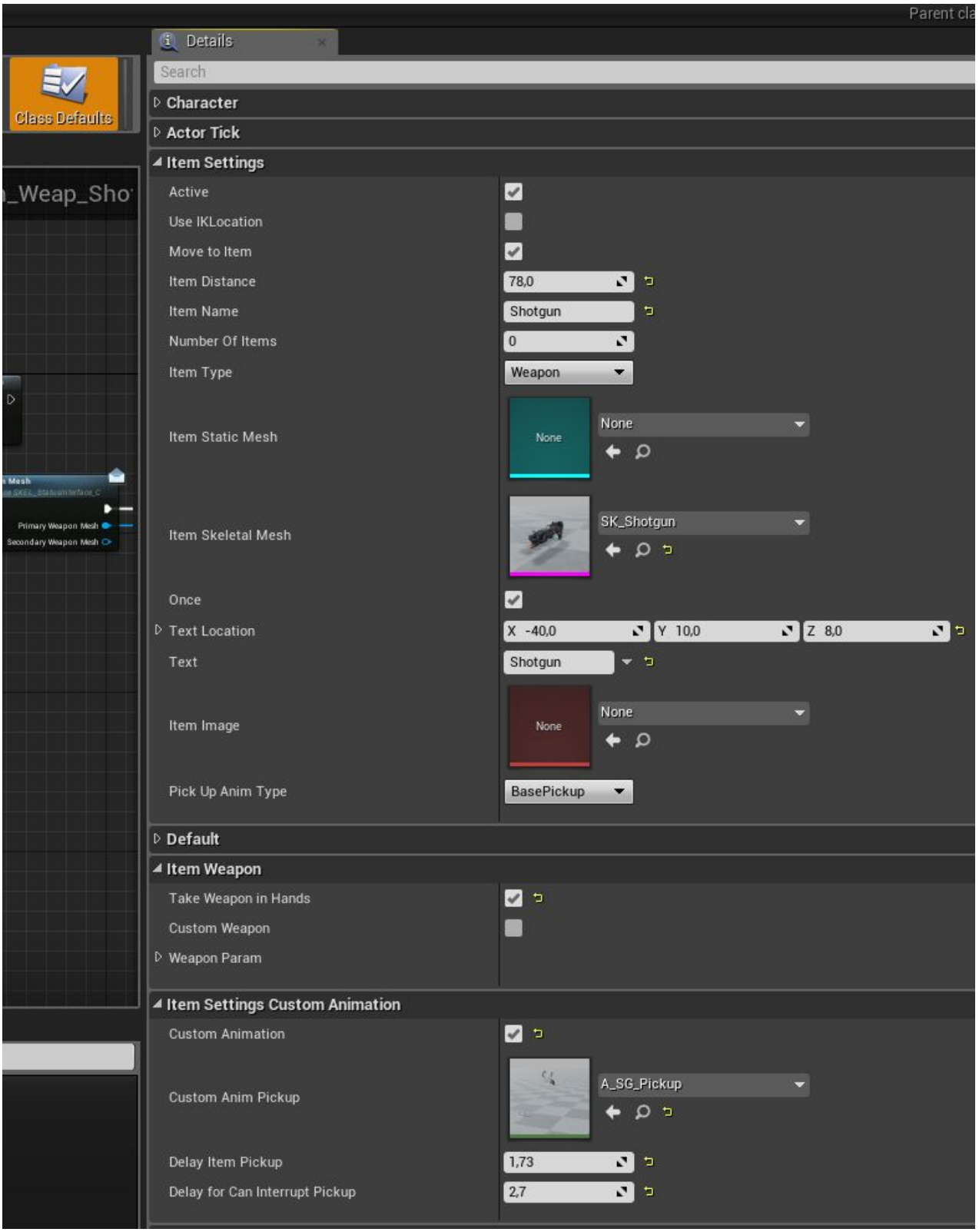
За предметы в проекте отвечает класс подбирания предметов **BP_PickupItem** служит для интерактивного взаимодействия персонажа с предметами и работает по следующей схеме.

Рассмотрим предмет для огнестрельного оружия. Вы можете сразу расположить предмет на сцене и настроить его там (функционал подбираемого предмета позволяет полностью создать оружие без WeaponStorage компонента), но тогда вам придется либо скопировать этот предмет либо использовать его один раз. Я рекомендую создать дочерний класс предмета и настроить его для многоразового использования + так его будет проще добавлять на сцену через блупринт.
Для этого кликните правой кнопкой мыши на **BP_PickupItem** далее “**Create Child Blueprint Class**”.



В проекте уже присутствуют заготовки дочерних классов для предметов в папке Items (оружие, патроны, канистра с бензином и пр..)

Откройте ваш новосозданный предмет перейдите в “**Class Defaults**” базовые настройки класса и в первую очередь приступим к блоку настроек “ItemSettings”. Разберем предмет на примере дробовика.



Active - Активны ли объект (можно ли его подобрать). Так как нам нужно подобрать дробовик оставляем включенным. Данный функционал может пригодится в тех. случаях когда предмет не нужно поднимать, например предмет за стеклянным шкафом и пока шкаф закрыт предмет должен иметь выключенный параметр Active
Use IKLocation - использовать IK положения рук на объекте.
Move To Item - будет ли персонаж двигаться к объекту. Так как мы будем подбирать дробовик со стола то оставляем включенным.
Item Distance - служит для ориентира у анимации. Подбор двуручного оружие анимировался при условии что персонажа от предмета находится на 78 см.
Number Of Items - кол-во предметов, т.к. это оружие то можно не указывать оно одно.Но если бы вы создавали предмет патронов или какого-нибудь ресурса то

нужно указывать кол-во.
ItemType - тип предмета в инвентаре, так как это оружие выбираем “Weapon”
ItemStaticMesh - Выбор внешнего вида в виде статичного меша, наш дробовик является скелетал меш поэтому оставляем пустым.
ItemSkeletalMesh - Выбор внешнего вида в виде скелетал меша, наш дробовик является скелет мешем поэтому выбираем нашу модель дробовика.
Once - подбирается ли предмет единожды. В нашем случае да. Это может быть полезно к примеру с ящиком патронов, подбирать патроны из которого можно несколько раз.
Text Location - Локальная локация размещения всплывающего текста.
Text - Текст который всплывает у предмета при приближении к нему персонажа.
ItemImage - *И*конка предмета при подбирании которая передаются в инвентарь. По умолчанию используется для патрон.
PickUpAnimType - тип анимации персонажа при подбирании предмета. анимация подбора предметов настраивается в компоненте “Interaction” в пешке.

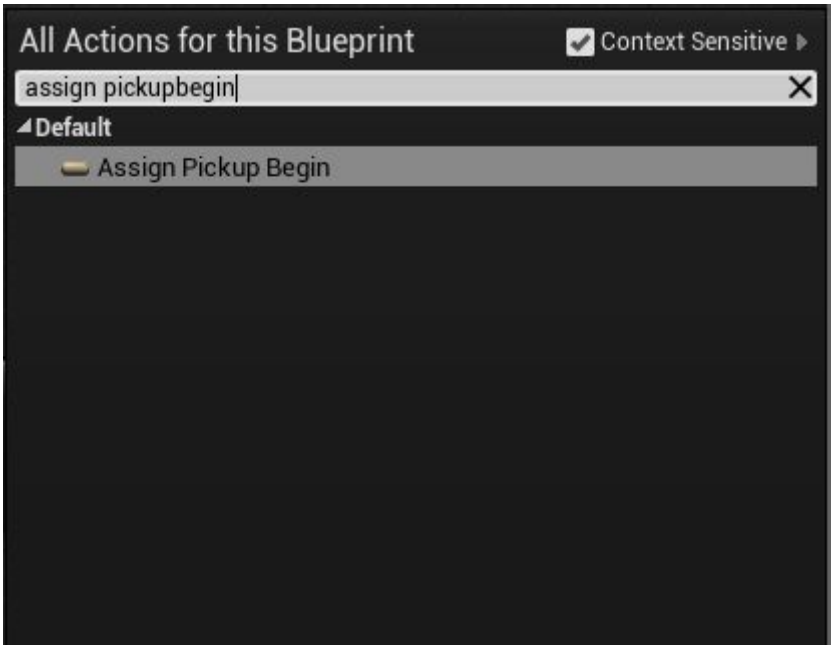
Стек настроек “Item Weapon”
С помощью этих натсроек вы можете выбрать будет ли браться оружие в руки (*TakeWeaponInHands*)или сразу пойдет в WeaponManager. Так как мы будем использовать специальную анимацию подбирания дробовика со стола то указываем что персонаж возьмет оружие в руки.
Параметр “*CustomWeapon*” позволяет создать уникальное оружие из предмета без добавления его в список “Weapon List” в компоненте “WeaponStorage”. Ниже идет структура настройки оружия - настройки описаны выше.

Стек настроек “Item Settings Custom Animation”
Данный стек настроек служит для замены анимации базового подбора. Так как у нас есть анимация подбора дробовика со стола то используем её.
Delay Item Pickup - Через какое время после начала подбирания предмета сработает логика в инвентаре (дробовик попадет в руки).
Delay for Can interrupt Pickup - через сколько времени после начала подбирания предмета можно прервать взаимодействие.

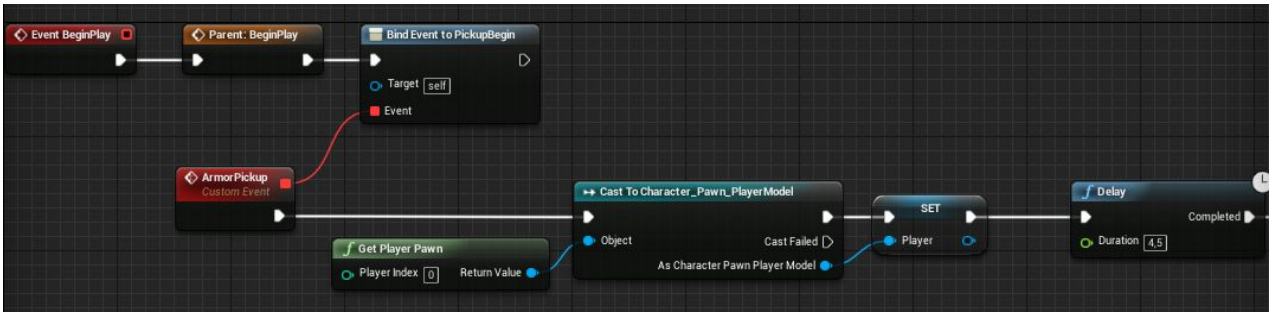
На этом наш предмет-дробовик готов - просто разместите его на сцену и подберите персонажем.

Предметы с дополнительной логикой.
Также вы можете создавать абсолютно уникальные предметы. В проекте есть примеры таких предметов это: бронежилет, противогаз.

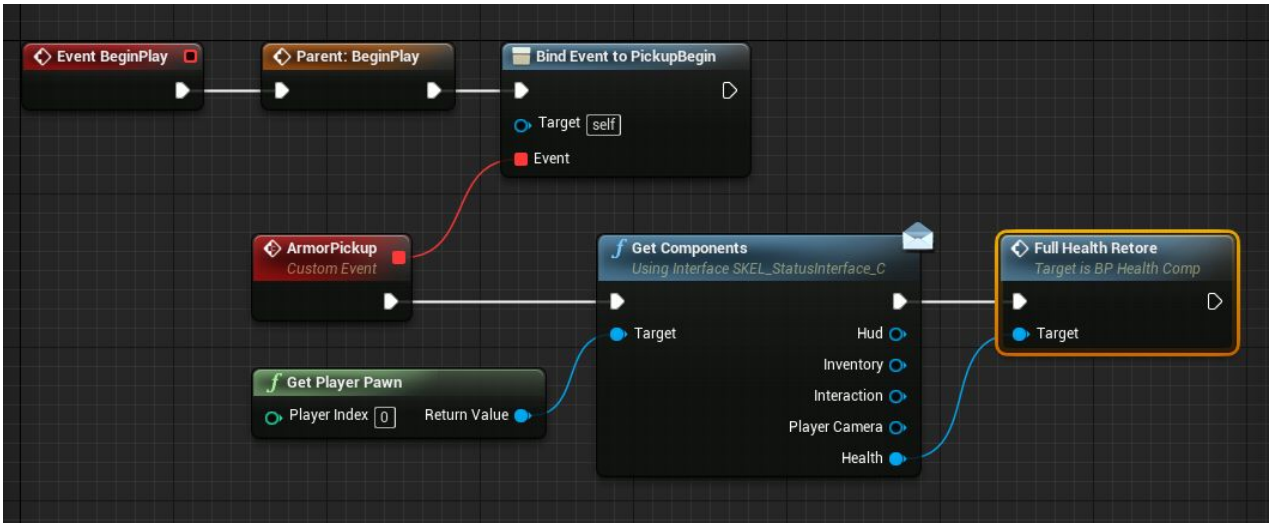
Создайте (как описано выше) новый предмет, укажите необходимые настройки. Далее перейдите в ивент граф и на ивент “BeginPlay” привяжите диспатчер “PickupBegin”.



В дальнейшем данный ивент сработает в момент начала подбирания предмета, у бронежилета это различные эффекты и увеличение здоровья у персонажа.



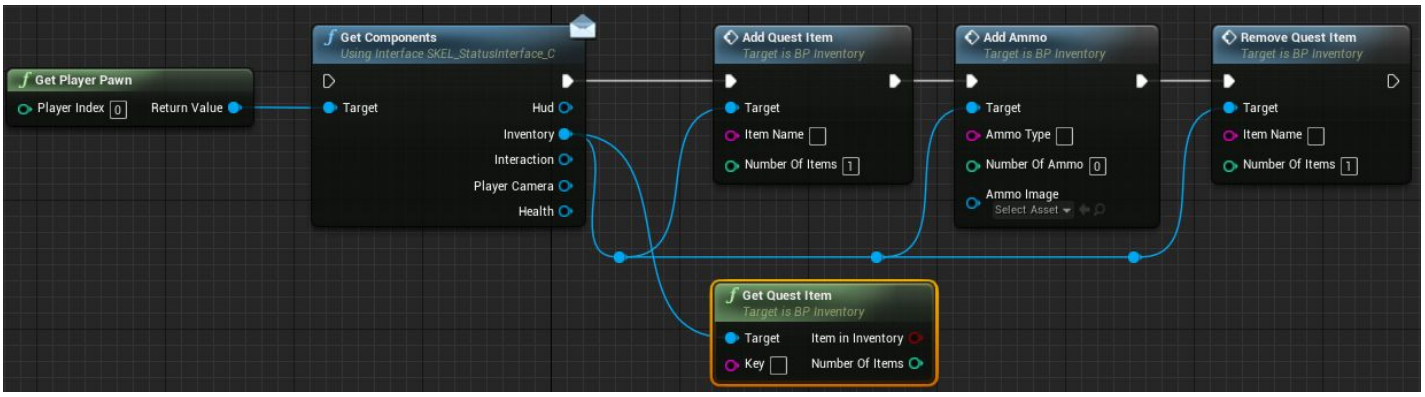
Вы можете создавать различные другие предметы, например аптечка выглядит так.



Предмет ссылается на компонент здоровья персонажа и запускает ивент полного восстановления здоровья. Интерфейс “Get Components” создан для быстрого доступа к компонентам персонажа.

Дополнительные ивенты в инвентаре

Для работы с инвентарем по умолчанию присутствуют следующие функции. Если вам нужно запустить логику не из пешки вы можете обратиться к инвентарю и к другим компонентам через интерфейс “Get Components”.



Add Quest Item - Добавляет в инвентарь квестовый предмет. Указываете имя и кол-во.
Get Quest Item - Запрос о наличии предмета в инвентаре. Например если в инвентаре есть ключ то можете открыть дверь и прочее применение.
Add Ammo - Добавить патроны указанного типа.
Remove Quest Item - Удалить предмет или его кол-во из инвентаря.

Опасные зоны

В проекте существует класс с помощью которого вы можете создавать опасные зоны для персонажей. Например газовые камеры, области в огне и прочее. Рассмотрим пример газового коридора из уровня “ZombieCity” где опасная зона наносит повреждение персонажам если у них нету противогаза.

Для создания опасной зоны просто перетащите на сцену класс **BP_DamageVolume** и растяните его по необходимой области.
Настройки:
Active - Активна ли опасная зона. Если активна она будет наносить повреждения.
Damage - Сколько повреждения будет наносится персонажу за определенный отрезок времени
DamageRate - Отрезок времени в секундах, через который будет наноситься

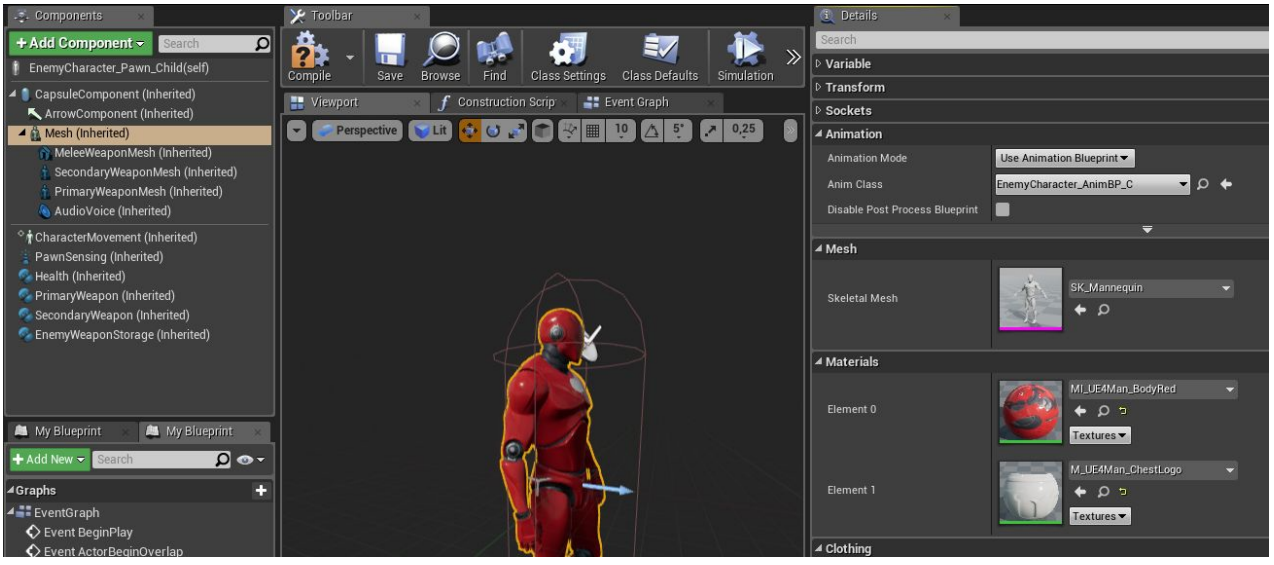
повреждение.
OnlyPlayer - Повреждение будет наноситься только игроку.
PlayerDoes'tGetDamage - Повреждение будет наносится всем кроме игрока.
DamageType - Тип повреждения. По данному типу вы сможете строить дополнительную логику к примеру вы можете создать тип повреждения “Огонь” и запускать эффекты горения у персонажей

Настройки “Protecteve Item”
UseProtectiveItem - *И*спользовать защитный предмет от данной опасной зоны.
ProtectiveItemName - *И*мя защитного предмета. Именно это имя предмета будет искать опасная зона в инвентаре персонажа.

Искусственный интеллект - Враги

В 3ем обновлении значительно расширились возможности искусственного интеллекта. В проекте присутствует базовый родительский класс врага - “**EnemyCharacter_Pawn**”. От данного класса создаются дочерние классы в виде типов врагов. В проекте по умолчанию присутствуют 3 типа: солдат, зомби и кибер зомби.

Рассмотрим настройки ИИ при создании нового врага. Создайте дочерний класс от класса “**EnemyCharacter_Pawn”** и откройте его. Для начала настроим внешний вид выделите компонент **Mesh** и выберете модель персонажа, вы также можете добавлять дополнительные меши или другие компоненты, например у врагов типа “Zombie” добавлены дополнительные меши которые делят персонажа на 3 части и подбираются случайным образом чтобы зомби не выглядели одинаково.

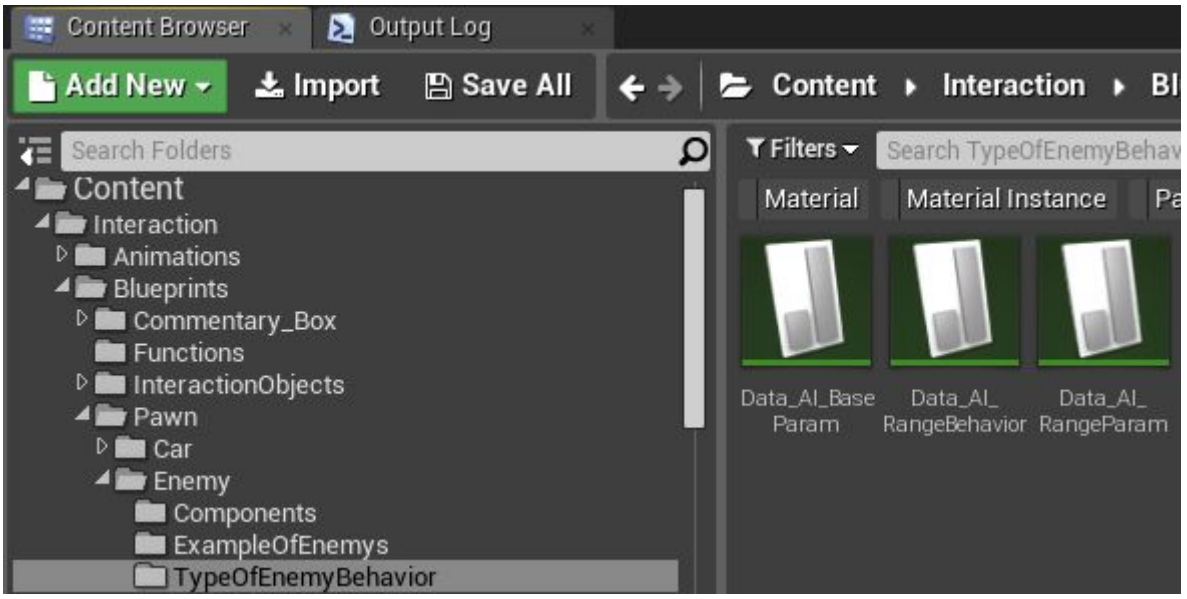


Далее настрйоте анимацию вашего нового врага для этого откройте “**EnemyCharacter_AnimBP**” и найдите категорию “**Animation**” в списке переменных. Выделяя различные структуры анимаций вы можете заменить текущую анимацию у персонажа.

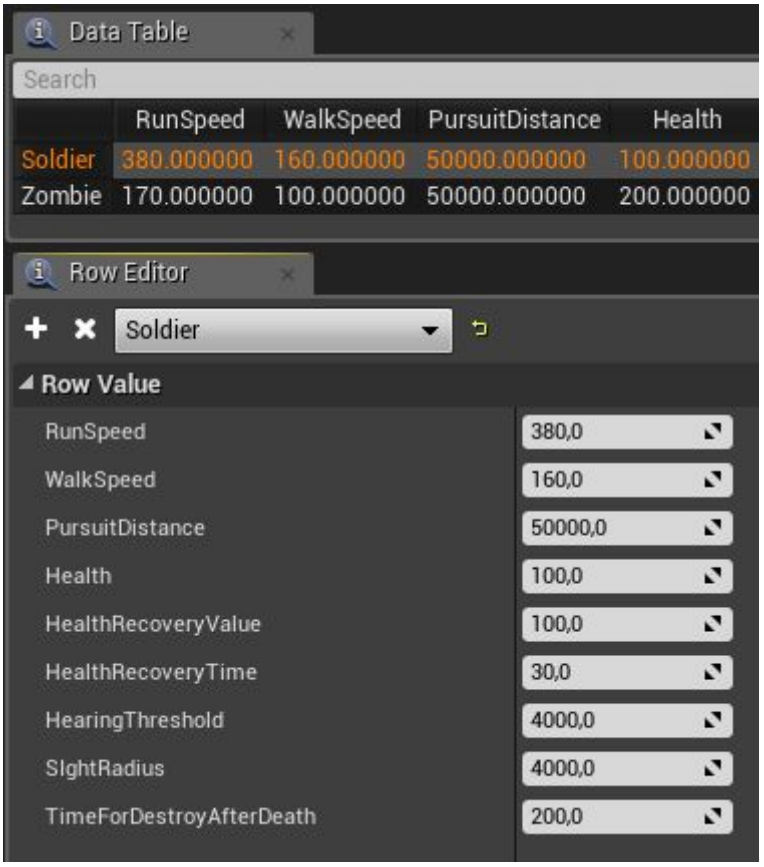
Внесите все необходимые изменения во внешний вид вашего врага и приступим к его настройкам. Первом делом настроим его базовые параметры.

Параметры врага

В проекте параметры врагам задаются при помощи дата таблиц. Она расположены здесь:



Откройте “**Data_AI_BaseParam**”



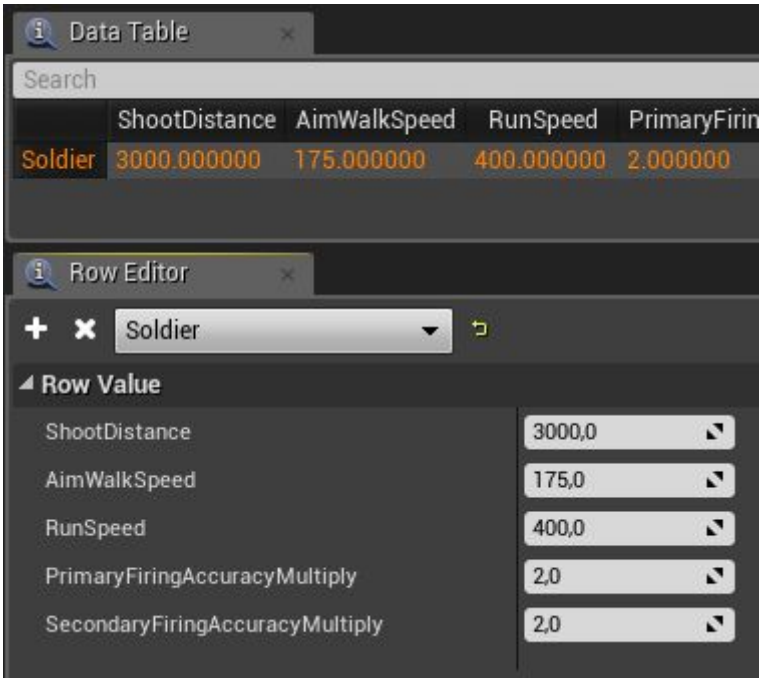
Для создания новой настройки базовых параметров кликните на + и введите имя параметров для вашего нового врага.

RunSpeed - Скорость бега персонажа
WalkSpeed - Скорость шага персонажа
PursuitDistance - Дистанция при которой персонаж будет продолжать преследование до цели во время боя.
Health - Базовое здоровье персонажа.
HealthRecoveryValue - Сколько может восстановить здоровья персонаж. Укажите 0

если персонаж должен восстанавливать здоровье полностью.
HealthRecoveryTime - Время за которое персонаж восстанавливает здоровье после повреждение. Укажите 0 если персонаж не должен восстанавливать здоровье.
HearongTheshold - расстояние на которое персонаж может слышать.
SightRadius - расстояние на которое персонаж может видеть.
TimeForDestroyAfterDeath - Время через которое персонаж будет удален после смерти (оптимизация). Укажите 0 тогда персонаж не будет удален.

Указав базовые настройки персонажа, переходим к следующим настройкам. Если ваш персонаж умеет использовать оружие дальнего боя (в том числе любые броски, стрельбу и прочее) то нужно указать настройки поведения при ведении дальнего боя.

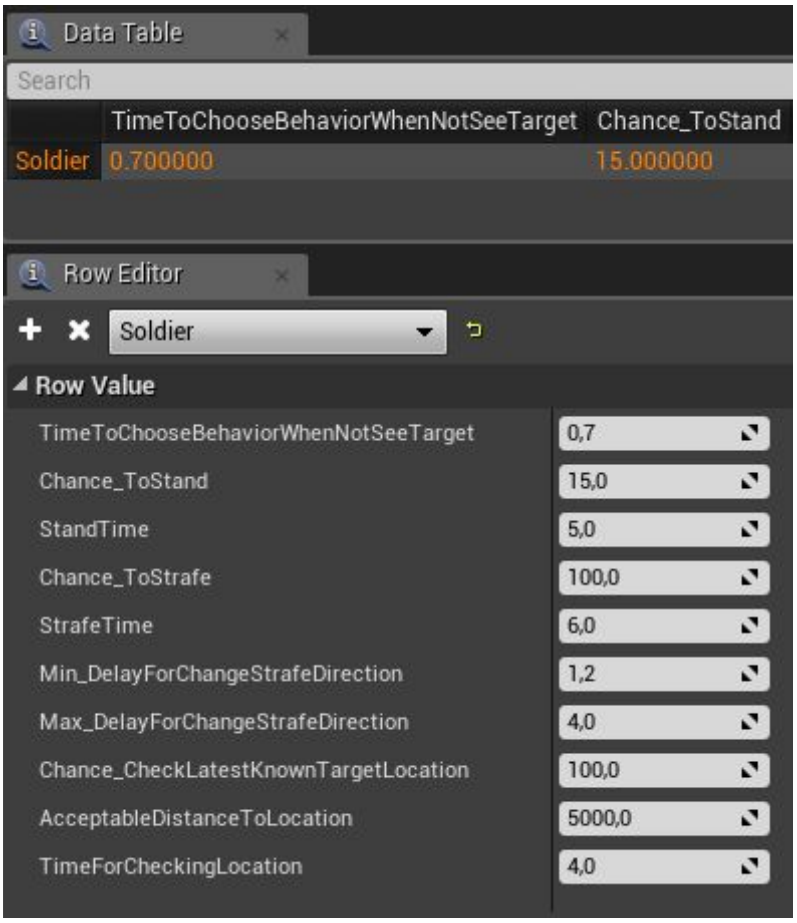
Откройте таблицу “**Data_AI_RangeParam**” добавьте новые настройки и дайте им имя (для удобства стоит задать тоже имя что вы указывали в базовых настройках)



ShootDistance - Расстояние до цели при котором враг может вести дальний бой.
AimWalkSpeed - Скорость персонажа при ведении дальнего боя.
RunSpeed - Скорость бега персонажа в боевом режиме.
PrimaryFiringAccuracyMultiply - Множитель точности первичного оружия от базовых его настроек.
SecondaryFiringAccuracyMultiply - Множитель точности вторичного оружия от базовых его настроек.

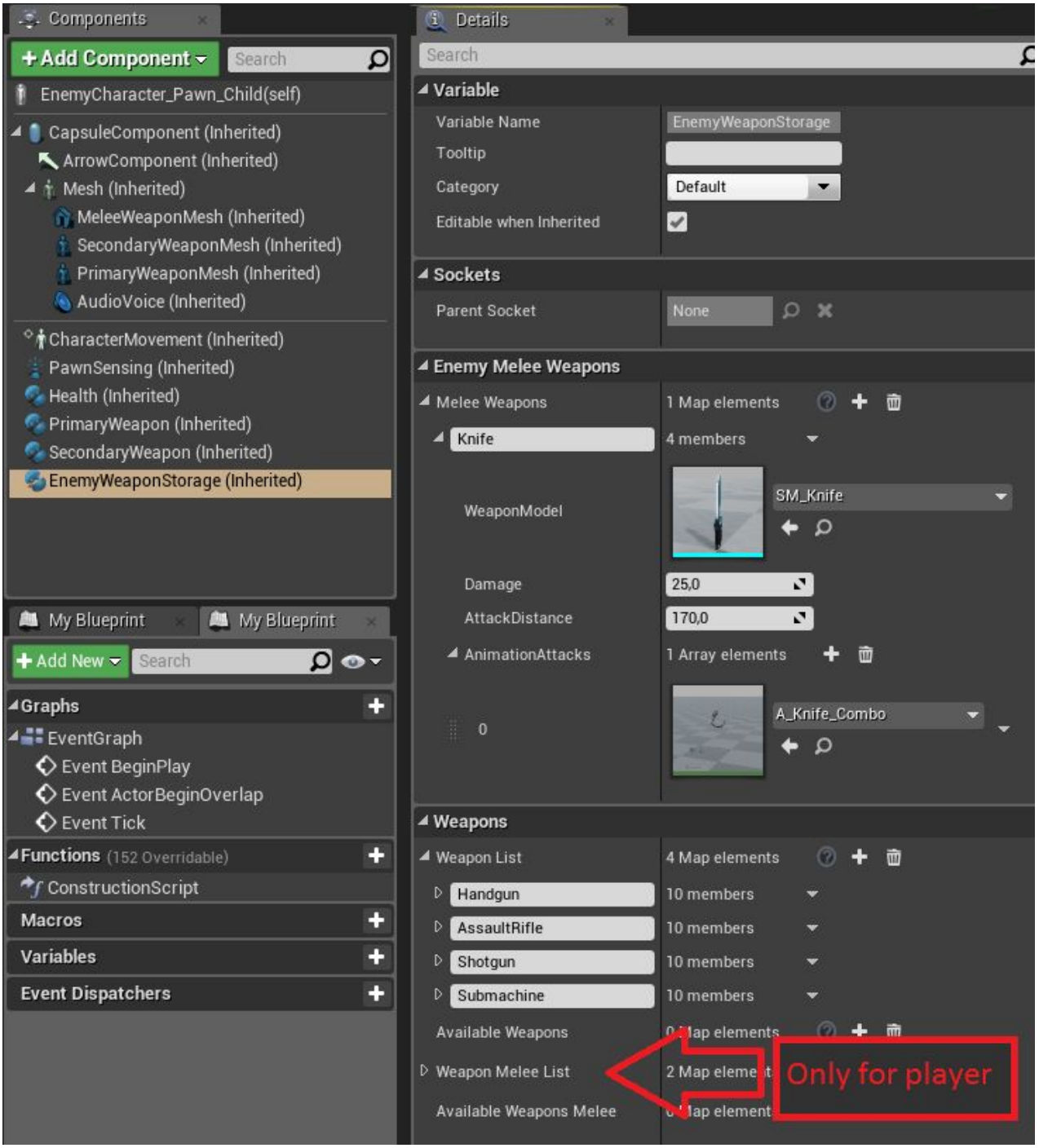
Указав базовые настройки дальнего боя приступим к настройкам поведения при ведени идальнего боя.

Откройте таблицу “**Data_AI_RangeBehavior**” добавьте новые настройки и дайте им имя. В данной таблице указываются настройки поведения врага при ведении дальнего боя.



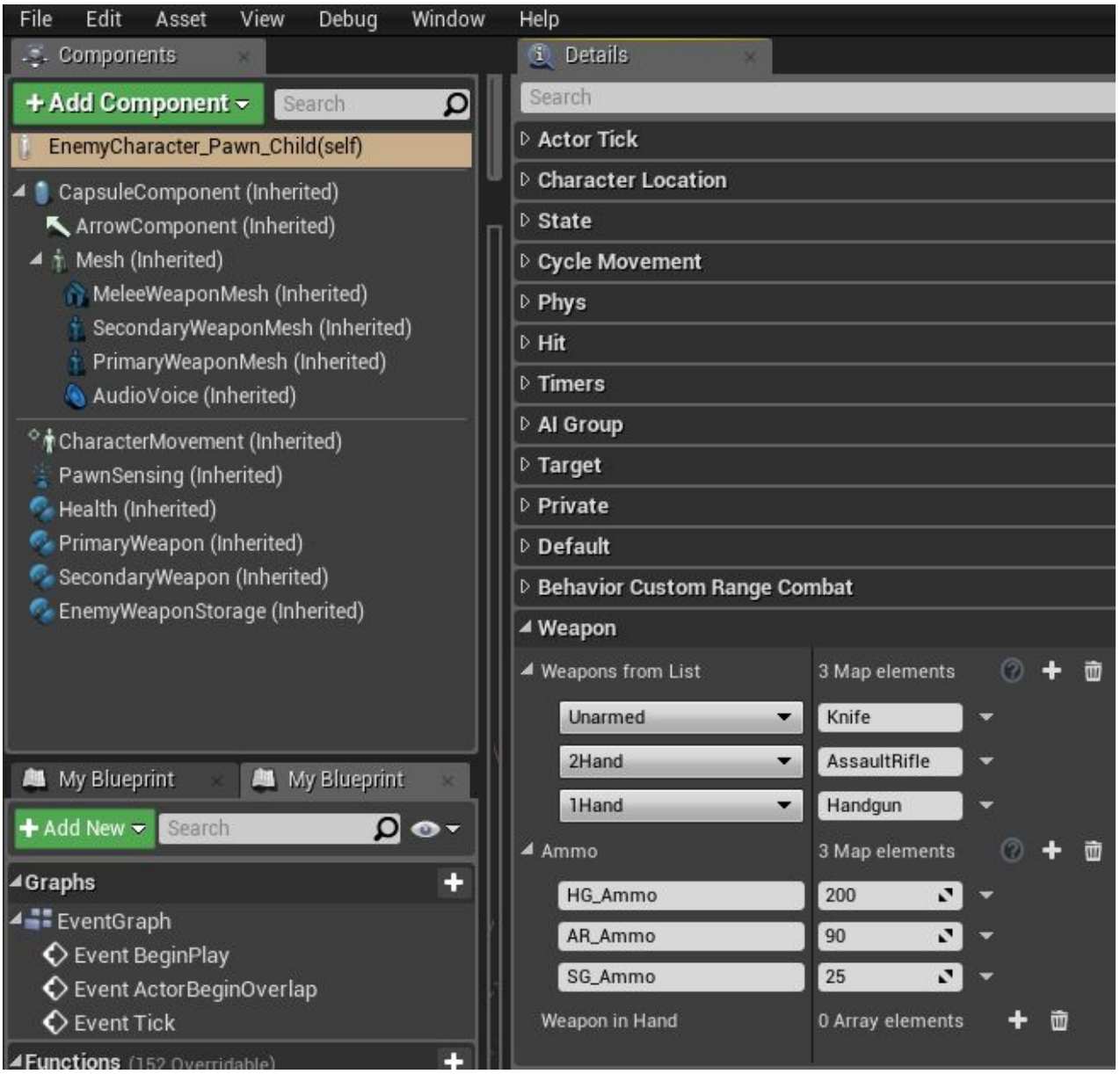
TimeToChooseBegaviorWhenNotSeeTarget - Время после которого персонаж вернется к патрулированию или начнет проверять последнюю известную ему позицию цели, если цель не в поле зрения.
Chance_ToStand - Шанс вести бой стоя на месте (15%)
StandTime - Если выпал шанс стоять на месте, то сколько времени будет стоять персонаж.
Chance_ToStrafe - Шанс при котором персонаж начнет использовать движения боком при стрельбе.
StrafeTime - Время движения боком
Min_DelayForChangeStrafeDirection - минимальный отрезок времени через который персонаж сменит направление движения боком.
Max_DelayForChangeStrafeDirection - максимальный отрезок времени через который персонаж сменит направление движения боком.
Chance_CheckLatestKnownTargetLocation - шанс проверить послдние известное место цели если цель пропала из поле зрения.
AcceptableDistanceToLocation - расстояние при котором персонаж пойдет обыскивать последнюю известную локацию цели.
TimeForCheckingLocation - Время обыскивания последней известной локации.

Следующим шагом стоит вооружить нового врага. Это может быть как оружие дальнего боя так и оружие ближнего боя. Откройте вашего созданного врага и перейдите в компонент “**EnemyWeaponStorage**”.



Оружие врагам добавляется по такому же принципу как и игроку за исключением оружия ближнего боя. Моменты нанесения повреждения указываются в самой анимации атаки врага. Внимание тут добавляется **все оружие** которое может использовать данный враг.

При помощи нотифай “**EnemyAttack**”.
После того как вы добавили все возможное оружие врагу, выберем с каким оружием он будет попадать на уровень. Вы можете всегда менять эти параметры при спавне врага или на сцене выделив его и перейдя в настройки в панели “**Details**”. Для выбора оружия по умолчанию в данном классе выделите корневой компонент в списке компонентов и справа в настройках найдите категорию “**Weapon**”.

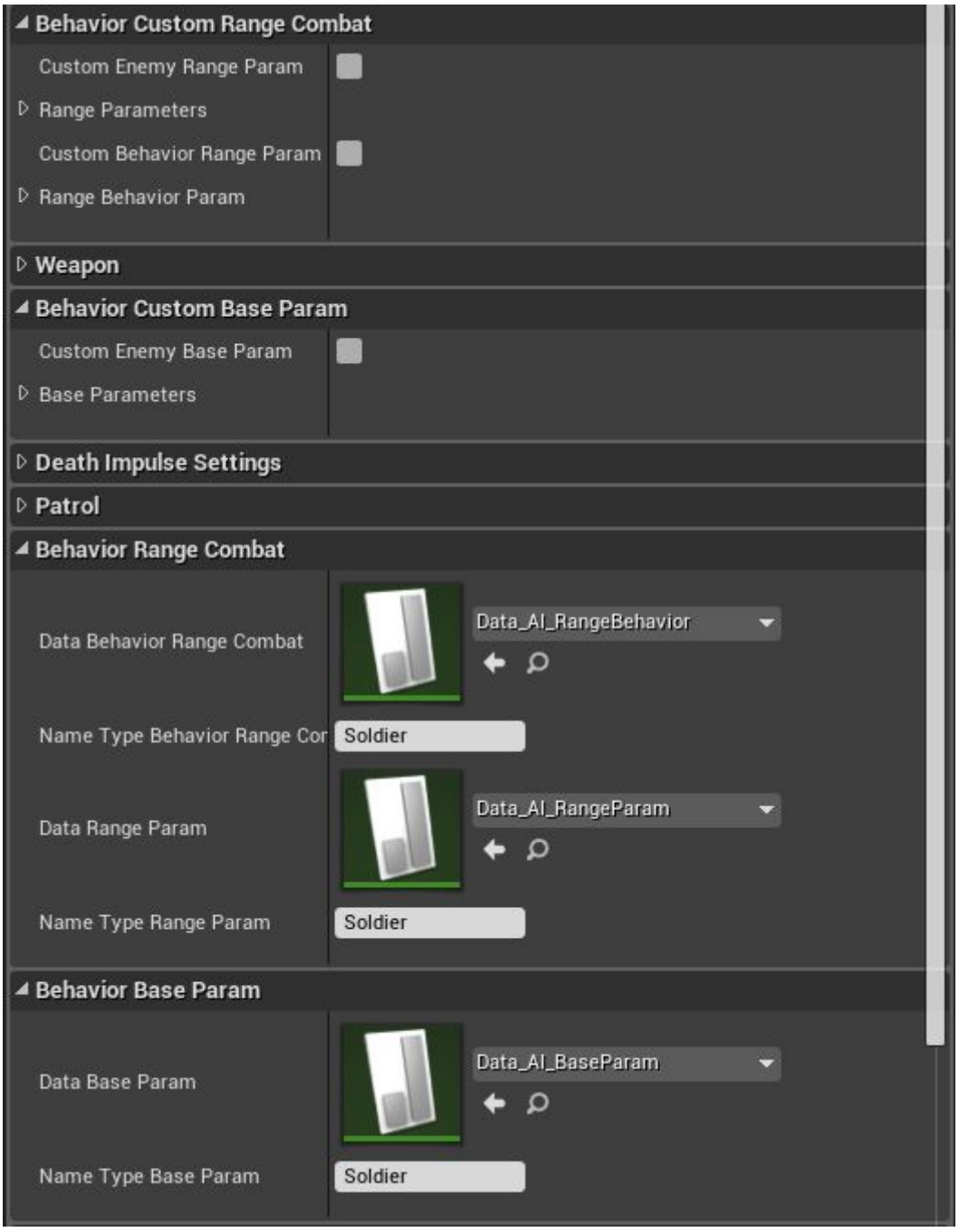


По умолчанию в проекте враги как и игрок могут использовать два оружия и оружие ближнего боя. В данной категории укажите оружие которое будет по умолчанию у данного врага, а также патроны к нему. Как только закончатся патроны враг перейдет в ближний бой.

Дополнительные параметры врага

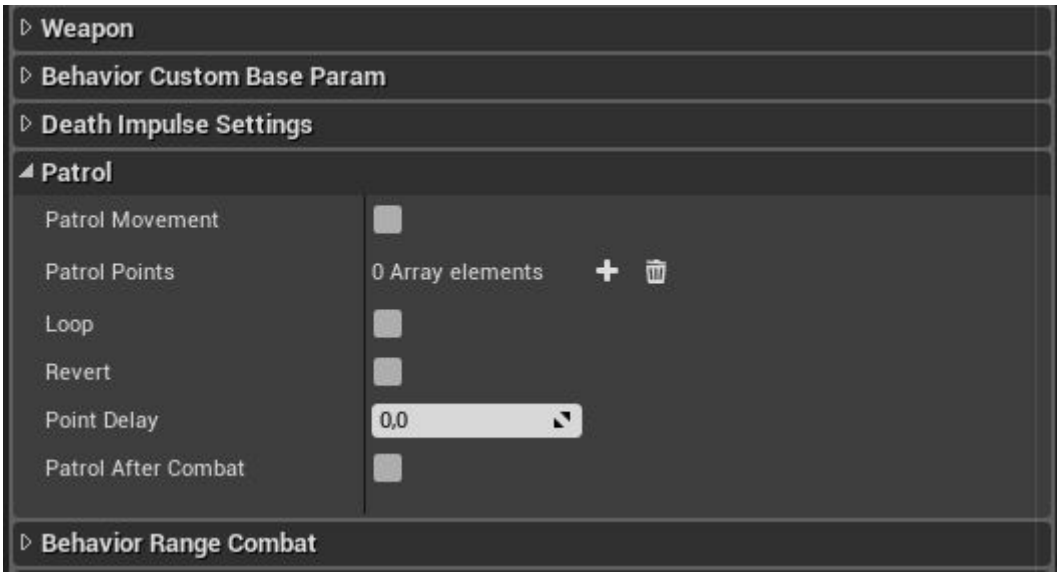
Так же рассмотрим все дополнительные параметры врагов которые вы можете настроить по умолчанию.

Категории поведения:



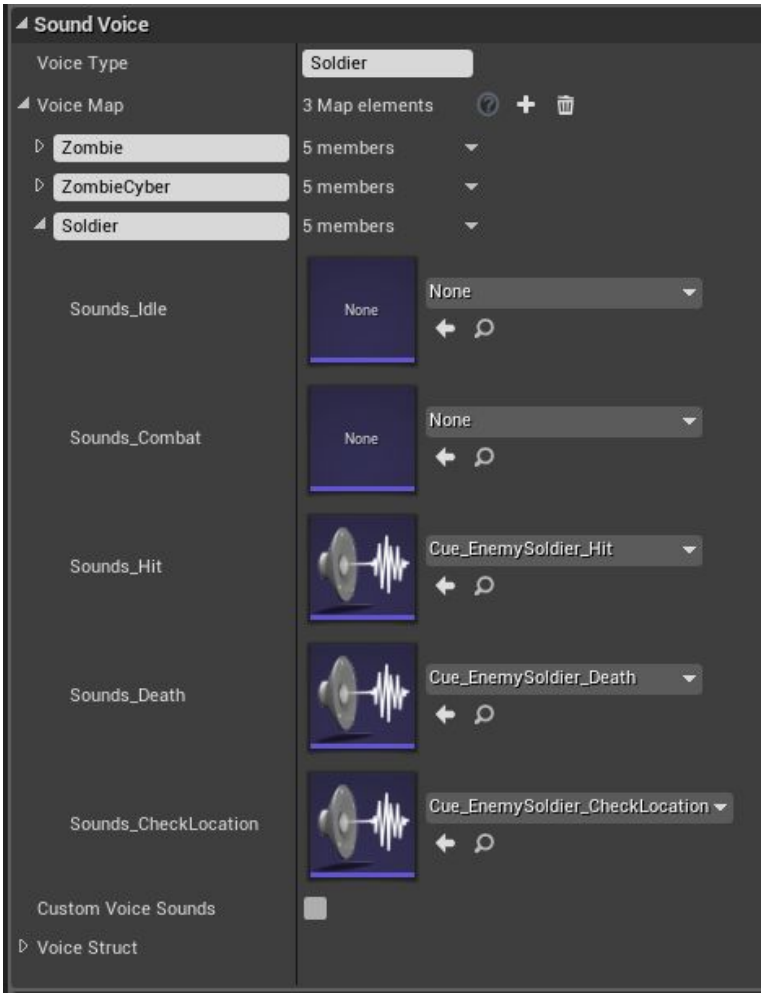
В данных категориях указывается имя настроек из дата таблиц. (настройки которые вы указали выше). Также в категориях с пометкой “**Custom**” вы можете задать настройки напрямую без использования таблиц.

Категория патрулирование:



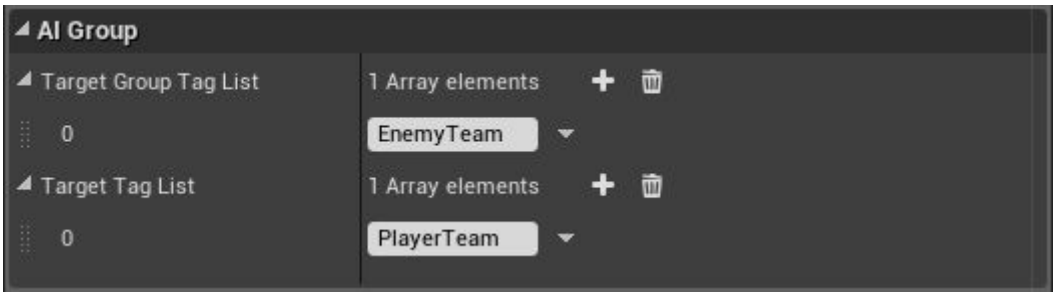
Patrol Movement - Будет ли изначально враг патрулировать область.
Patrol Points - Локации (точки) для маршрута патрулирования.
Loop - После достижения последней точки маршрута начнет ли враг обходить территорию заново.
Revert - Если параметр включен то враг при возобновлении патрулирования вернется в первую точку по точкам маршрута по которым прошел.
PointDelay - Время которое проведет враг по достижений точки маршрута.
PatrolAfterCombat - Возобновит ли патрулирование враг как только выйдет из боя.

Настройка голоса:



В данной категории вы можете задать настройки голоса врага, рычания в простаивающем состоянии как у зомби, звуки голоса получения повреждения, звуки смерти, звуки проверки локации цели. Вы можете создать сразу тип голосов, например Soldier подойдет ко все людям и просто указывать в параметре *VoiceType* - Soldier.

Категория AI Group



Враги могут иметь как несколько типов врагов так и несколько типов союзников. В данной категории указывается к какой группе относится враг и какие группы для него являются враждебными.

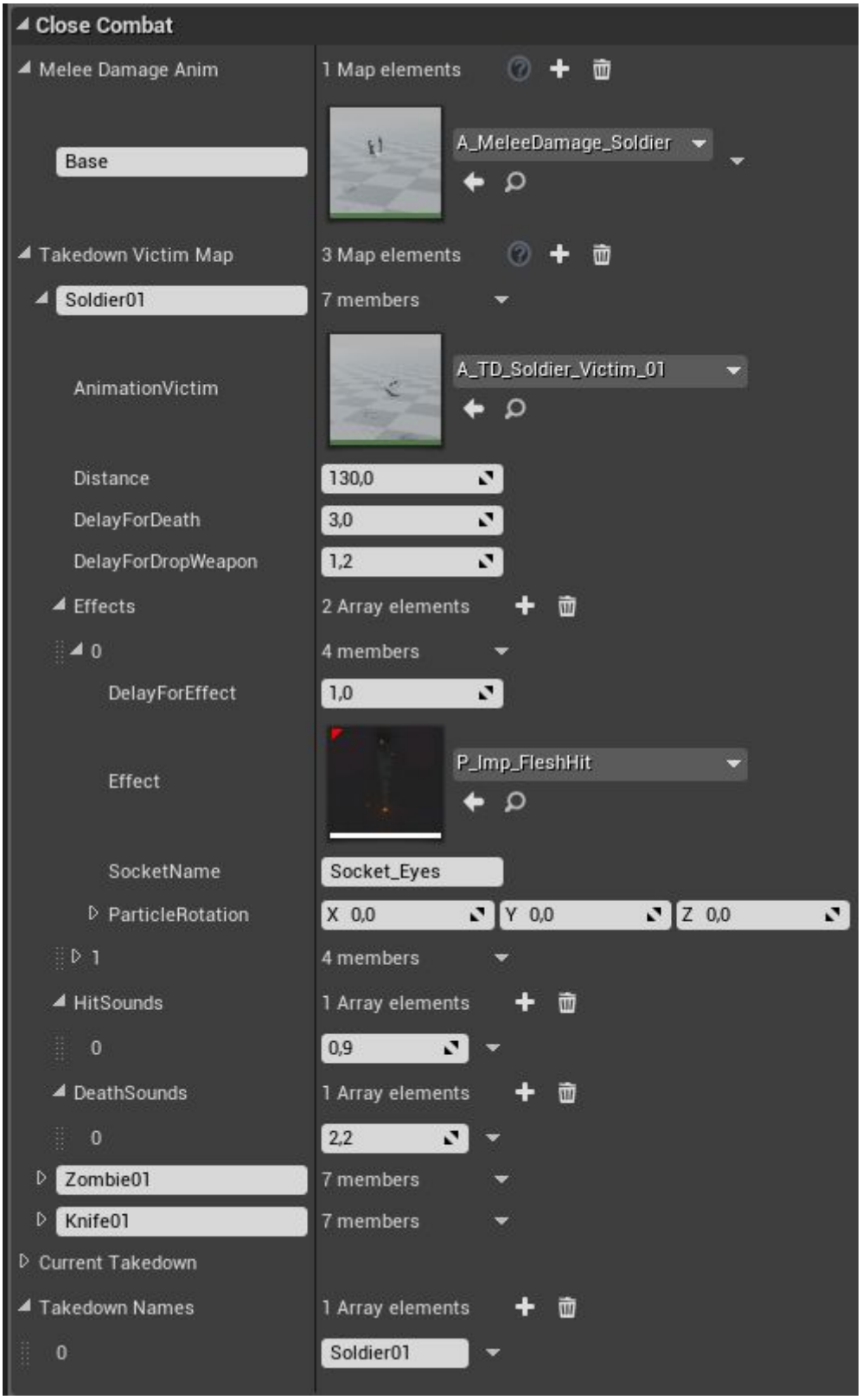
Категория Weak Spots

В данной категории указывается слабые месте врага при попадании в которые увеличивается урон. По умолчанию служит для попаданий в голову, также вы можете использовать данный функционал к примеру при создании боссов с уязвимыми местами.
UseWeakSpots - *И*меет ли данный персонаж уязвимые места.
Bone for Weap Spots - список костей являющимися слабыми местами у врага.
Weak Spot Damage - кол-во повреждения наносимого при попадании по слабым местам. При значении равным 0 отнимется все здоровья врага, что приводит к его смерти.

Ближний бой с врагами.

Враги могут получать урон от оружия ближнего боя а так же игрок может добивать или убивать врага синхронизированными анимациями при помощи функционала “**Takedown**”.

Раскройте категорию “**CloseCombat**”:

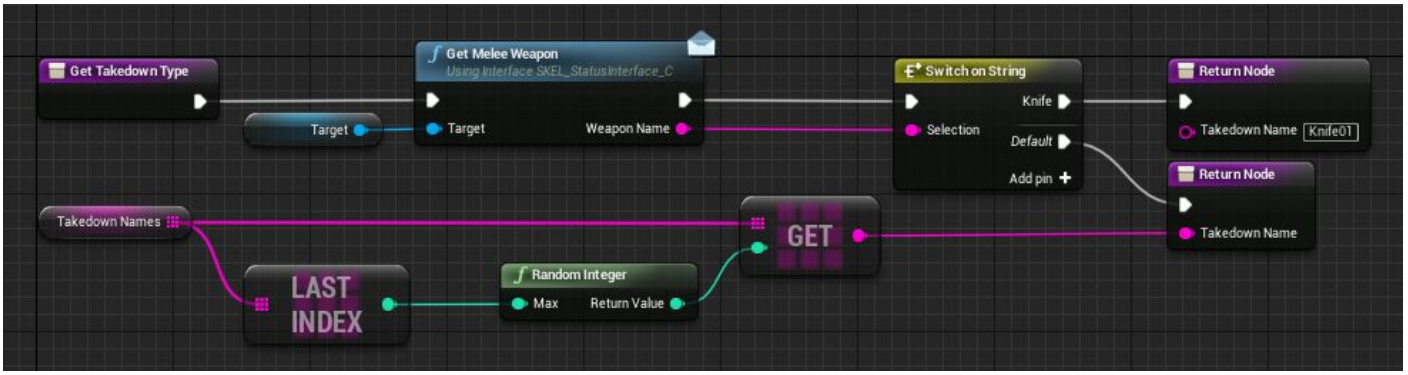


Melee Damage Anim - содержит в себе набор анимации для получения повреждений. По умолчанию добавлена одна анимация для каждого типа врагов. Тип анимации при ударе указывается в оружии ближнего боя у игрока. Указывайте имя тип анимации и непосредственно саму анимацию.

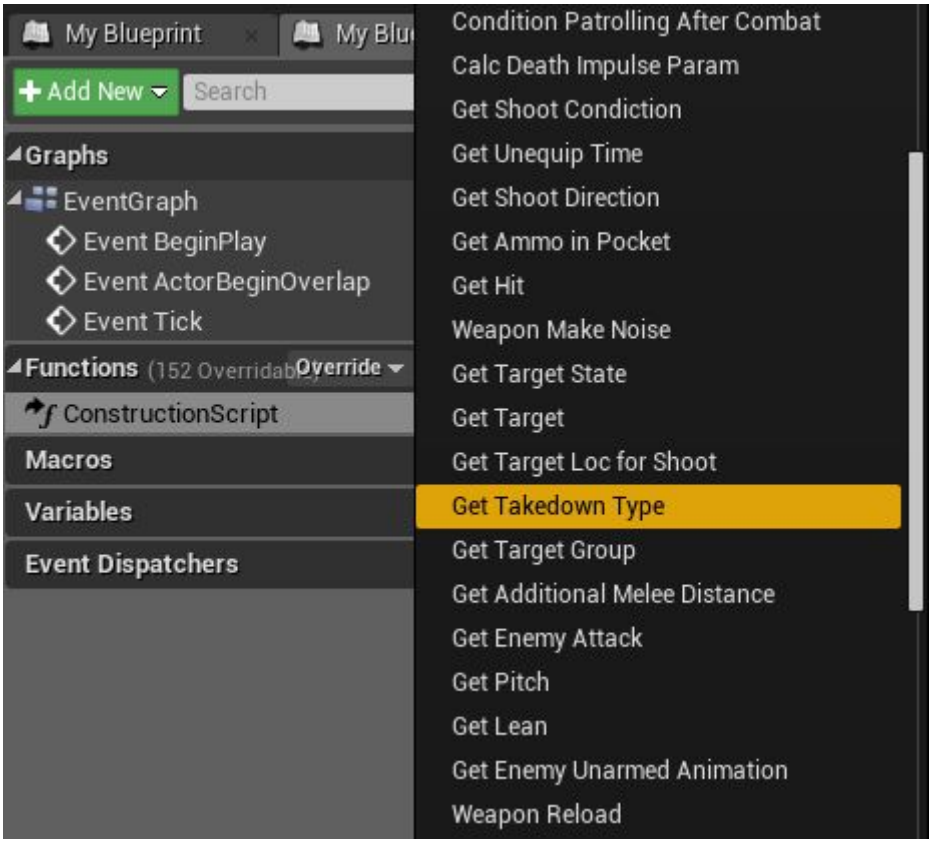
Takedown Victim Map - Настройки добивания врага (Takedown), в данной группе указываются доступные для данного врага добивания.
AnimationVictim - Анимация добивания у жервы (врага).
Distance - Расстояние на котором должны располагаться персонажи во время добивания (атакующий и жертва).
DelayForDeath - Задержка после которой жертва умирает.

DelayForDropWeapon - Задержка после которой жертва выронит оружие.
Effects - содержит в себе эффекты (партиклы) которые будут проигрываться во время добивания. Вы можете указать неограниченное количество эффектов.
DelayForEffect - Задержка после которой проигрывается эффект.
Effect - Выбор эффекта (партикл).
SocketName - локация эффекта.
ParticleRotation - ротация эффекта.
HitSounds - массив проигрывания звуков голосового получения урона, сами звуки указываются в категории “**SoundVoice**”.
DeathSounds - массив проигрывания звуков смерти, сами звуки указываются в категории “**SoundVoice**”.
Takedown Names - поддерживаемые данным врагом добивания указанные выше.

Таким образом настраивается добивания у врагов. По умолчанию условия для добивания это сближения вплотную с противником. Условия указываются в функции “**GetTakedownType**” и выглядит она так.

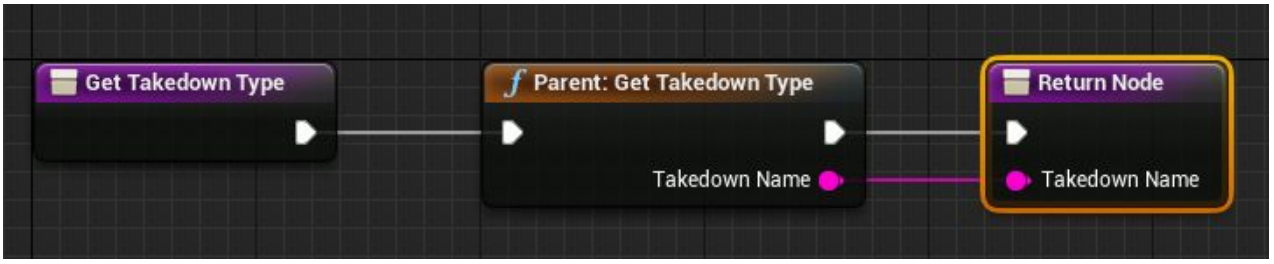


Выбирается любое добивание из списка “*Takedown Names”,* но если цель вооружена ножом то выбирается добивание с ножом. Чтобы изменить условия для данного игрока перепишите данную функцию следующим образом:

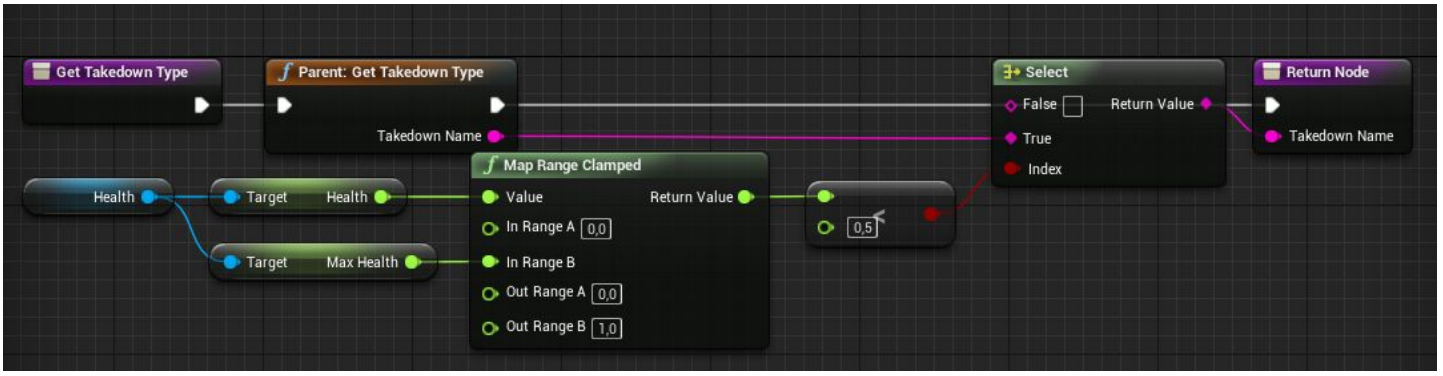


Нажмите на кнопку “Override” в списке функций и найдите там фйнкцию “*Get Takedown Type*”.

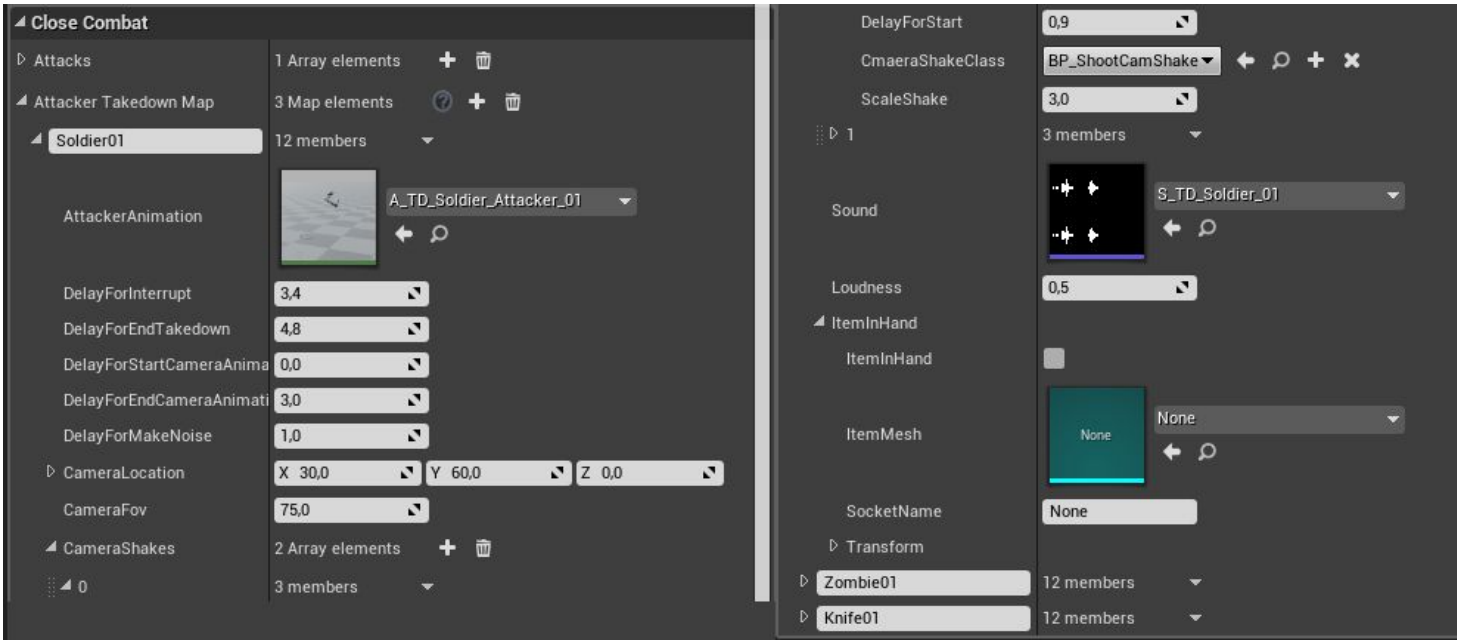
Откройте функцию и вы увидите следующее:



Узел “*Parent:Get Takedown Type*” это учет базовых условий (описано чуть выше) от родительского класса. Вы можете добавлять сови любые другие условия. Например такие, что добить противника можно только если у него меньше 50 процентов здоровья.



Оставьте переменную на выходе из функции “*Takedown Name*” пустой и тогда врага будет невозможно добить.
Это касается настроек у врага, теперь настроем добивание у игрока. Для этого перейдите в пешку игрока и выделите компонент “Interaction”, далее перейдите в категорию “CloseCombat”.



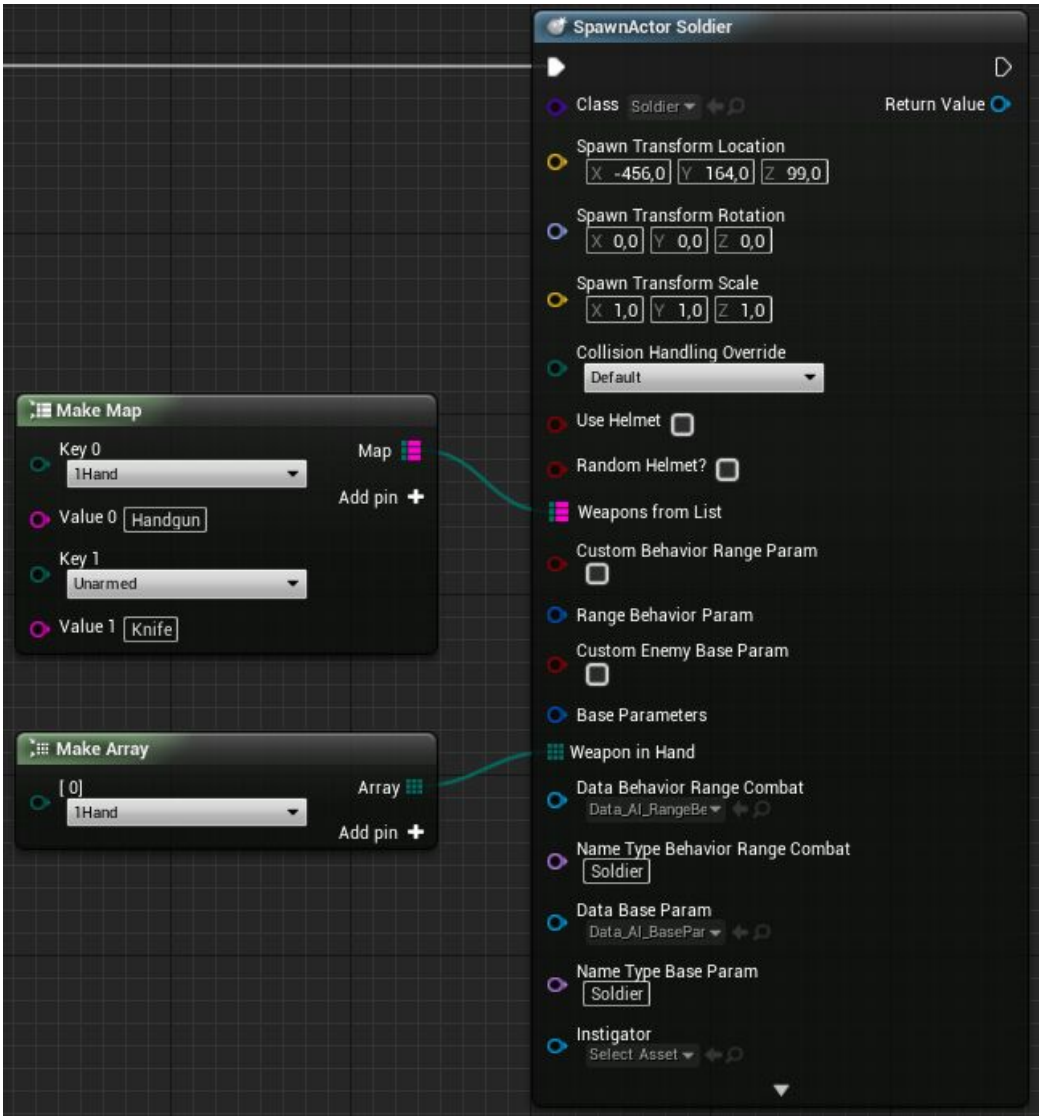
AttackerTakedownMap - По аналогии с жертвой данный массив содержит в себе натсройки атакуещего при добиваниях.
Обязательно укажите тоже самое имя для атакующего добивания, что вы указывали в настройках жертвы.
AttackerAnimation - Анимация добивания для атакующего.
DelayForInterrupt - Задержка после которой можно прервать добивание.
DelayForEndTakedown - Время окончания добивания.
DelayForStartCameraAnimation - Время начало анимации камеры.

DelayForStartCameraAnimation - Время завершения анимации камеры.
DelayForMakeNoise - Время когда добивание создаст шум бля ИИ. (рядом стоящие ии могут услышать).
CameraLocation - Локация камеры для анимации в это положение будет смещена камера при анимации во время добивания. Это socketOffset у springarm.
CameraFov - Field of view камеры при анимации
CameraShakes - Эффекты тряски камеры при нанесении ударов. Вы можете указать произвольное кол-во подобных трясок.
DelayForStart - Время тряски камеры (каждая последующая тряска камеры будет использовать задержку от задержки предыдущей)
CameraShakeClass - выбор класса тряски камеры.
ScaleShake - множитель силы тряски.
Sound - звук добивания.
Loudness - сила шума для слуха ии.
ItemInHand - Настройки предмета в руке. (аналогичные настройкам описанным выше для подбора предметов).

Таким образом вы можете добавлять различное число разных добиваний, включая специфические.

Добавление врагов на сцену через Blueprint

В проекте также учтена возможность добавления врагов на сцену при помощи узла “**Spawn Actor From Class**”:



Выберете в параметре “class” вашего врага и появятся дополнительные параметры врага, также укажите локацию где должен появится враг. Обязательно нужно указать, какое оружие будет использовать враг и какое оружие изначально у него в руках.

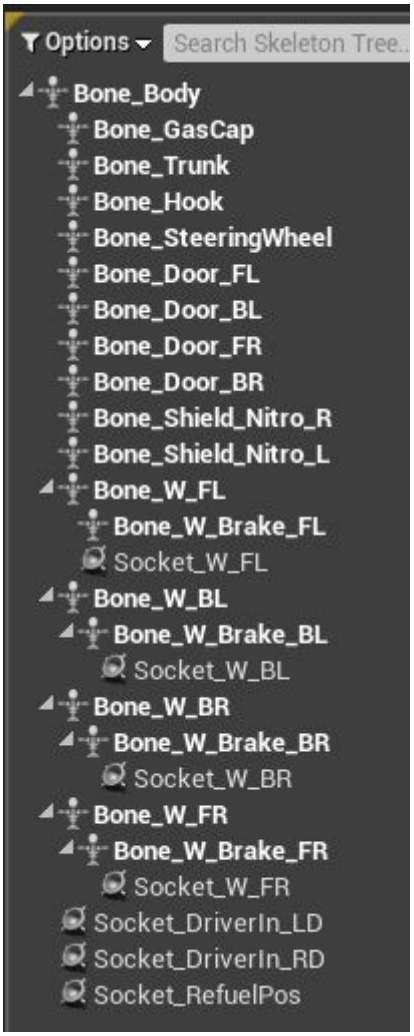
Для добавления дополнительных параметров рекомендовано использовать дочерние классы от класса “**EnemyCharacter_Pawn**”. Для примера у врага солдата реализовано поддержка шлема которая спасает его от одного попадания в голову (*UseHelmet RandomHelmet?*). У врагов типа “Zombie” реализовано рандомизированный внешний вид.

Машины

В проекте существует поддержка взаимодействия с транспортными средствами. Транспортные средства реализованы на основе класса “WheeledVehicle”. Транспортные средства содержат различные модули и могут тонко настраиваться в зависимости от геймплея.

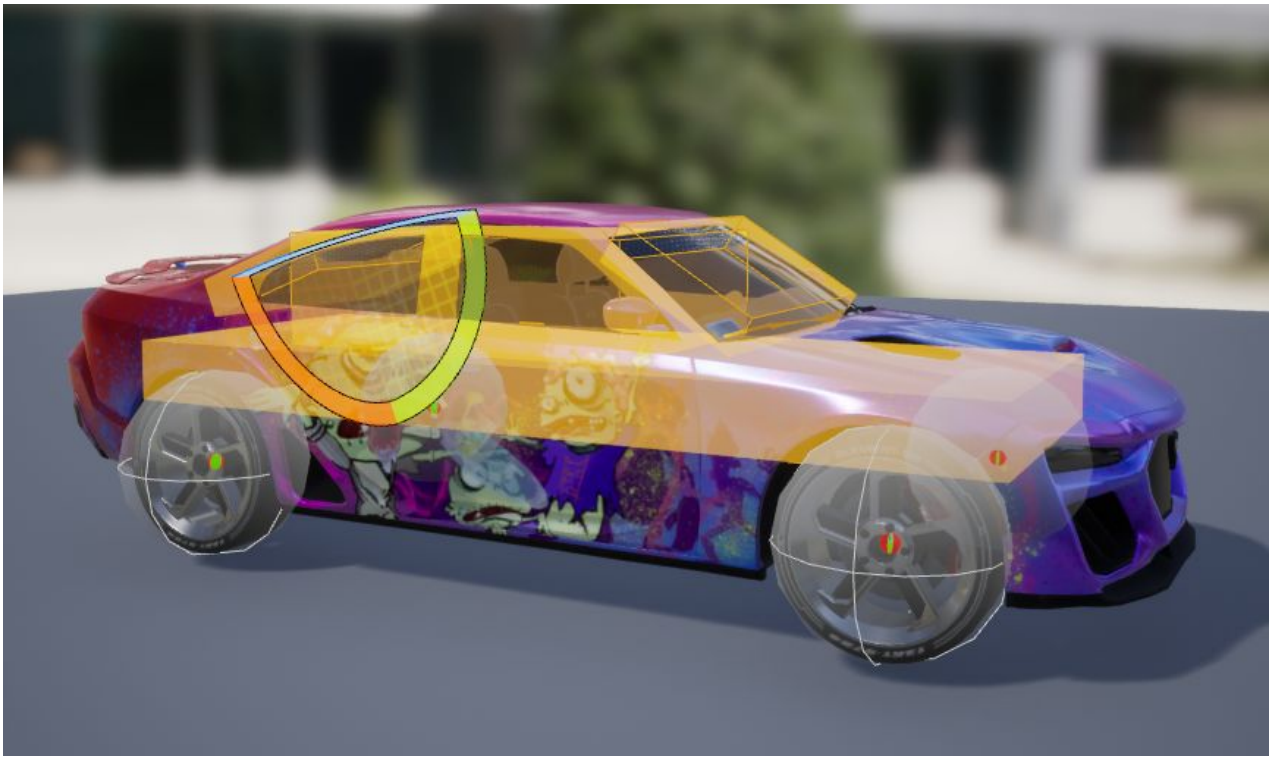
Добавление новой машины

Для начало создайте новую машину ка куказано здесь “Vehicle Art Setup”, но только укажите название костей как указано на изображении:

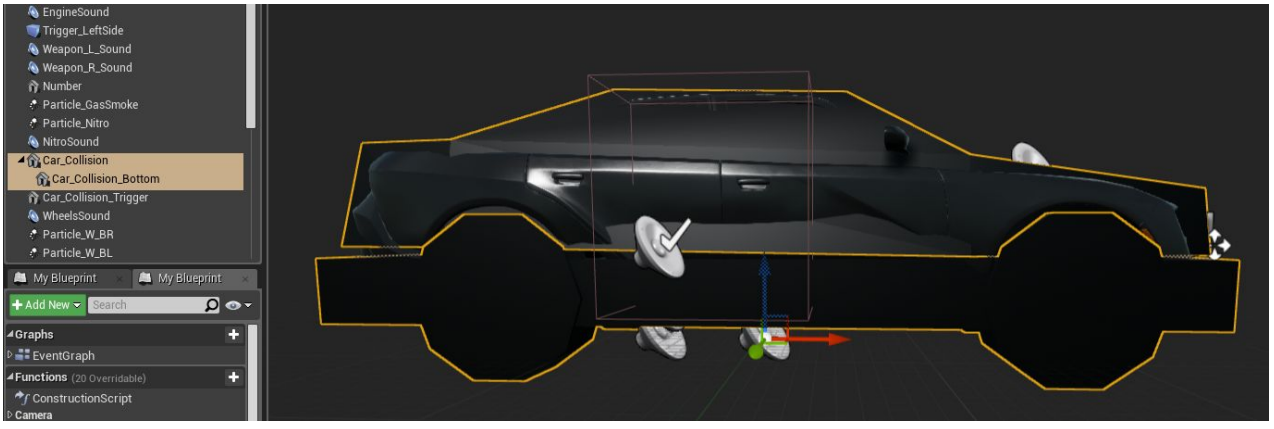


Рекомендую экспортировать машину “SyferJet” и использовать кости с нужным направлением, переместив их в положения колес и прочего вашей машины.
Кости которые должны иметь обязательные имена:

Bone_Body - Название кости для каркаса машины.
Bone_W_.. - Название колес машины.
Bone_W_Brake.. - Используете тоже название для тормозных колодок. (если таковые есть).
Остальные кости не имеет значения как вы назовете.
Добавьте обязательно следующие сокеты:
Socket_W_.. - сокеты для колес.
Socket_DriverIn_LD - Место с которого персонаж начнет открывать левую дверь.
Socket_DriverIn_RD - Место с которого персонаж начнет открывать правую дверь.
Socket_RefuelPos - Место откуда персонаж будет заливать бензин.
Следующим шагом добавьте коллизию в физическом ассете (phat) машины.
Коллизия должна быть довольно простой по аналогии с готовой машиной:



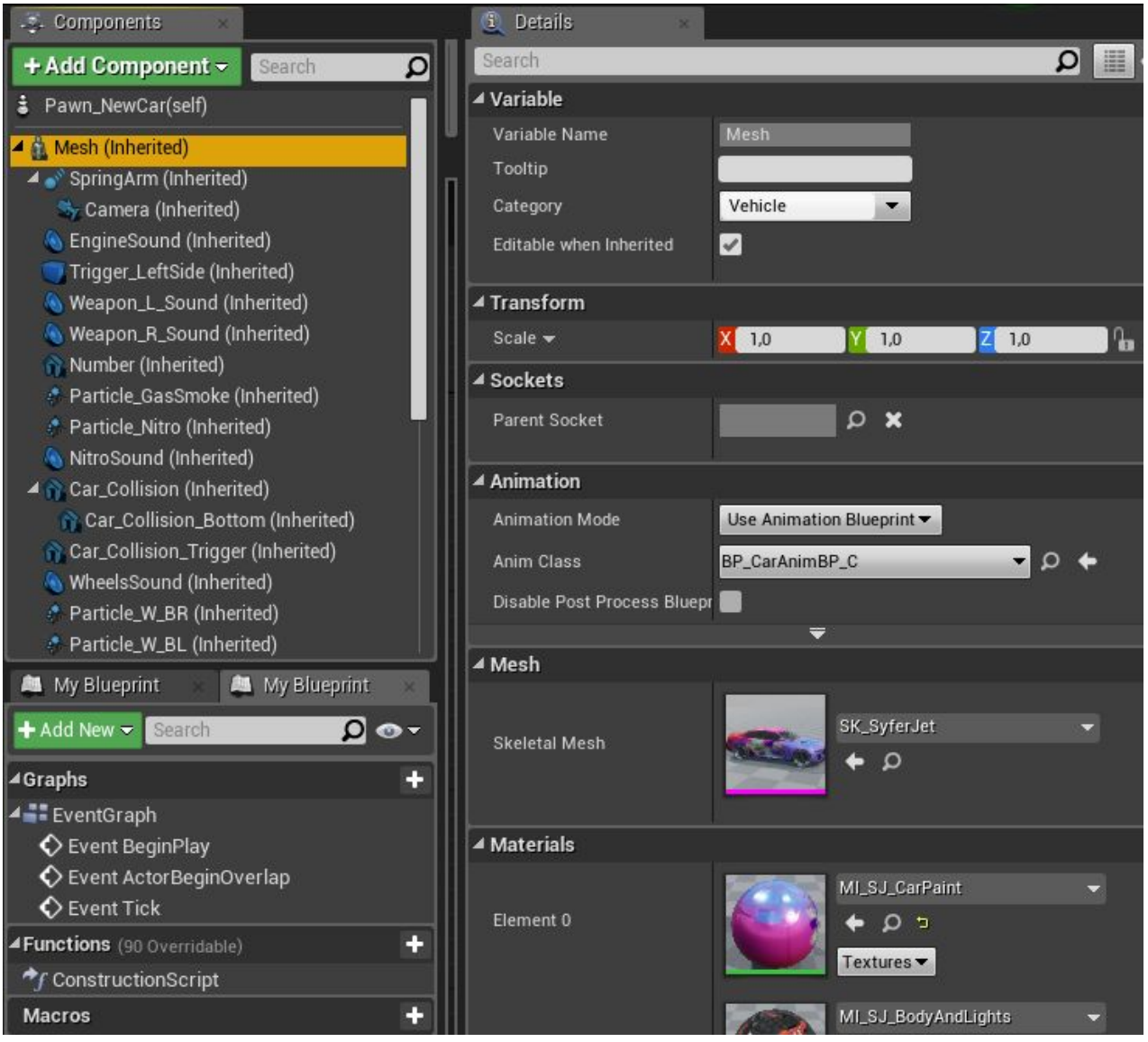
Так же добавьте 2 статик меша с верхней коллизией и нижней коллизией машины как это сделано у готовой машины:



Это нужно для того чтобы машина могла проскакивать мелкие кочки или поребрики, нижняя коллизия активируется когда персонажа выходит из машины.

Далее приступим к добавлению новой машины и все возможные настройки. В проекте реализована два типа транспорта - боевой и базовый. Боевая машина является дочерним классом от базовой машины и имеет расширенные настройки.

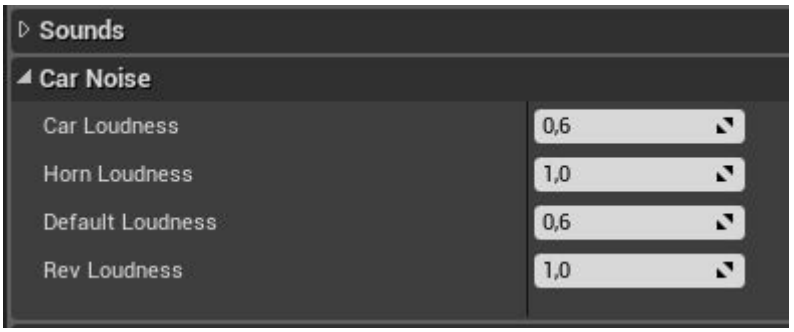
Создадим дочерний класс от боевой машины (**Pawn_Car_Combat**), чтобы рассмотреть все доступные по умолчанию настройки (Если ваша машина не должна поддерживать боевой функционал рекомендовано создавать дочерний класс от класса “**Pawn_Car**”).
Создав дочерний класс дайте ему имя (на данном примере указано Pawn_NewCar) и откройте его. В первую очередь добавьте вашу модель машины выделив компонент **mesh**:



Также вам нужно будет сделать ретаргет “**BP_CarAnimBP”** для скелета вашей машины и указать в “**AnimClass**” - внимание без **AnimBP** машина не будет анимироваться и соответственно корректно работать.

Настройки машины.
Технические параметры машины задаются в компоненте “VehicleMovement”, ознакомиться с параметрами можно здесь: Vehicle
Откройте вашу новую созданную машину и перейдите в базовые настройки:

Первом делом настройте парамтеры взаимодействия с машиной.
Char Relative Pos Driver - Локация водителя относительно машины. (Место водителя)
Car Anim Struct - В данном блоке указывается вся анимация машины при взаимодействии с ней персонажем.
Следующим шагом укажите все используемые звуки для вашей машины, а также шум от неё:

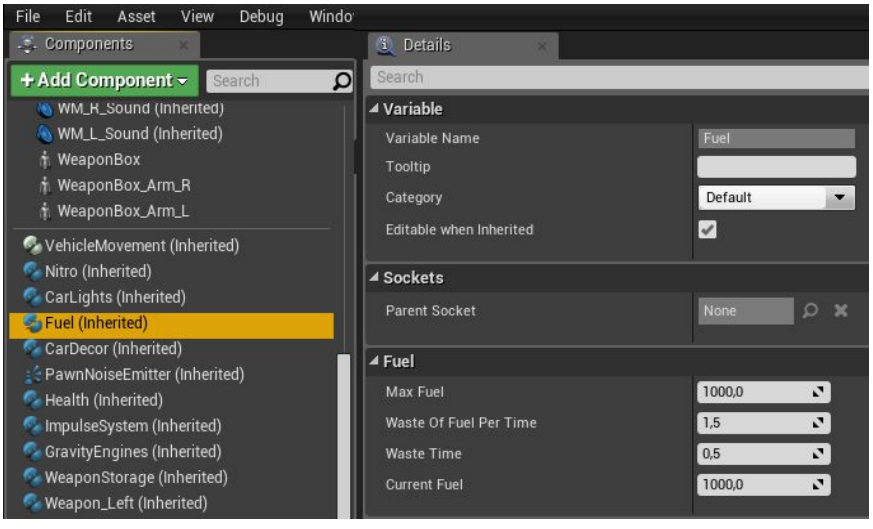


Модули машины

Рассмотрим все доступные модули для машины для этого перейдите в категорию “Car Modules”:

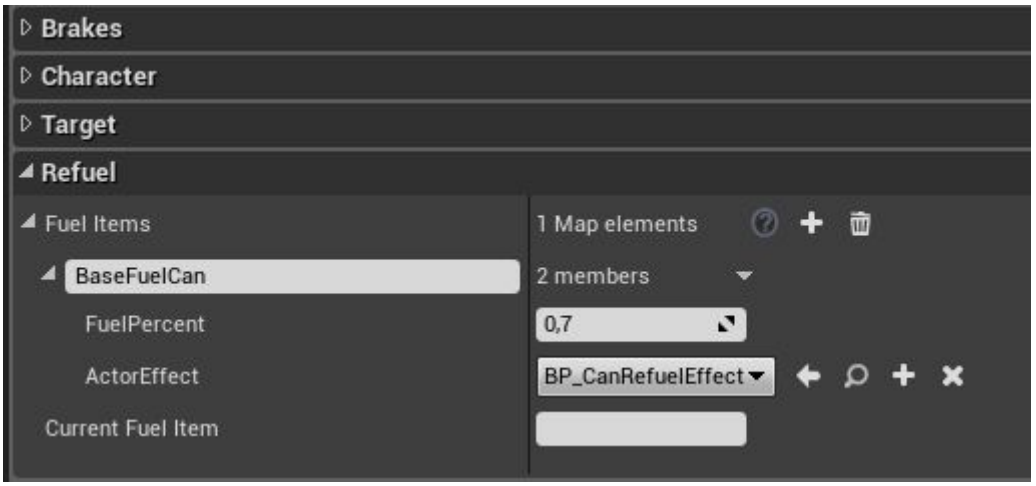
Рассмотрим для начала базовые модули:
CarLocked - будет ли заперта машина. (Этот параметр можно менять динамически).
Car Key Name - ключ который может открыть данную машину. Просто создайте предмет (pickup item) с указанным именем и если он будет в инвентаре то машину можно открыть.

UseFuel - При активном модуле машина начнет использовать горючее и при пустом баке машина заглохнет, настройки горючего задаются в компоненте “Fuel”.



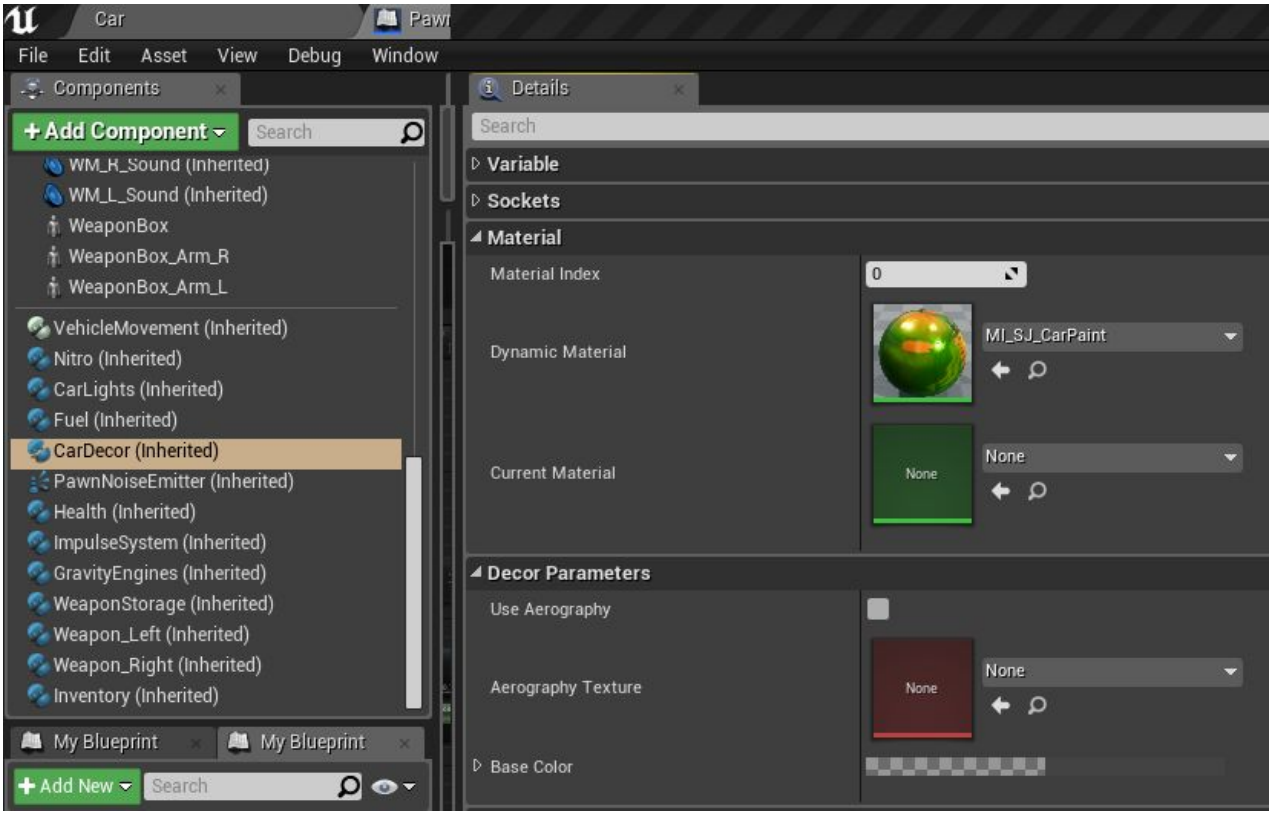
Max Fuel - Максимальное количество горючего.
Waste of Duel Per Time - Количество затраты горючего в промежуток времени.
Waste Time - Промежуток времени через который будет тратится горючее.
Current Fuel - Текущее кол-во горючего в баке.

Также вы можете создавать различные предметы которые могут заправлять вашу машину. Они указываются в базовых настройках в категории “Refuel”:



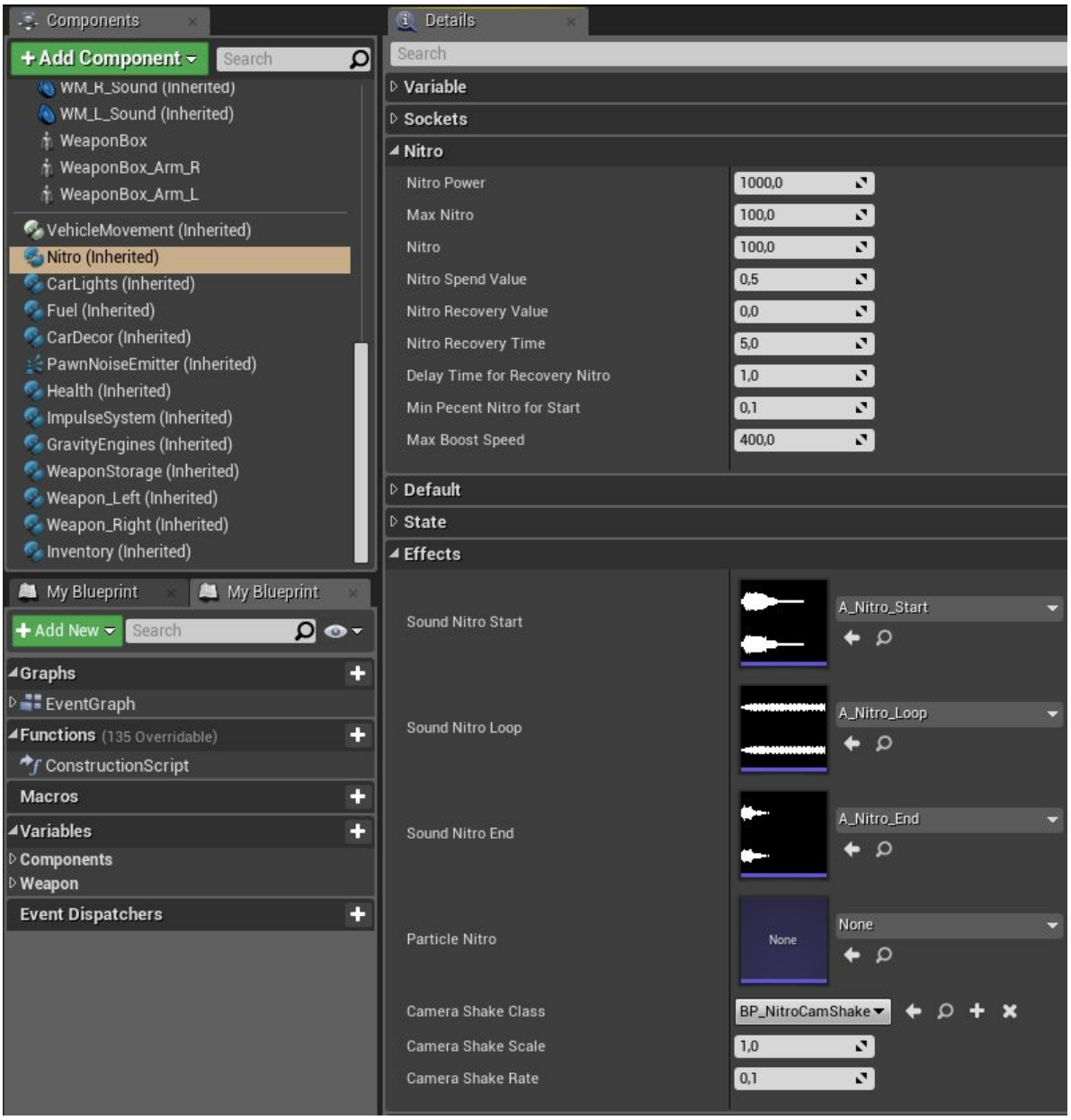
Fuel Items - предметы, которые могут заправить данную машину. Имя предмета в списке должно совпадать с предметом, который может быть в инвентаре.
FuelPercent - Сколько процентов горючего заправляет машину данный предмет.
ActorEffect - Класс содержащий в себе различные эффекты для заправки - по умолчанию это анимированная канистра со звуками и физикой.

Use Car Decor - модуль внешнего вида машины. Содержит в себе настройки аэрографии и цвета машины. Настройки указываются в компоненте “CarDecor”



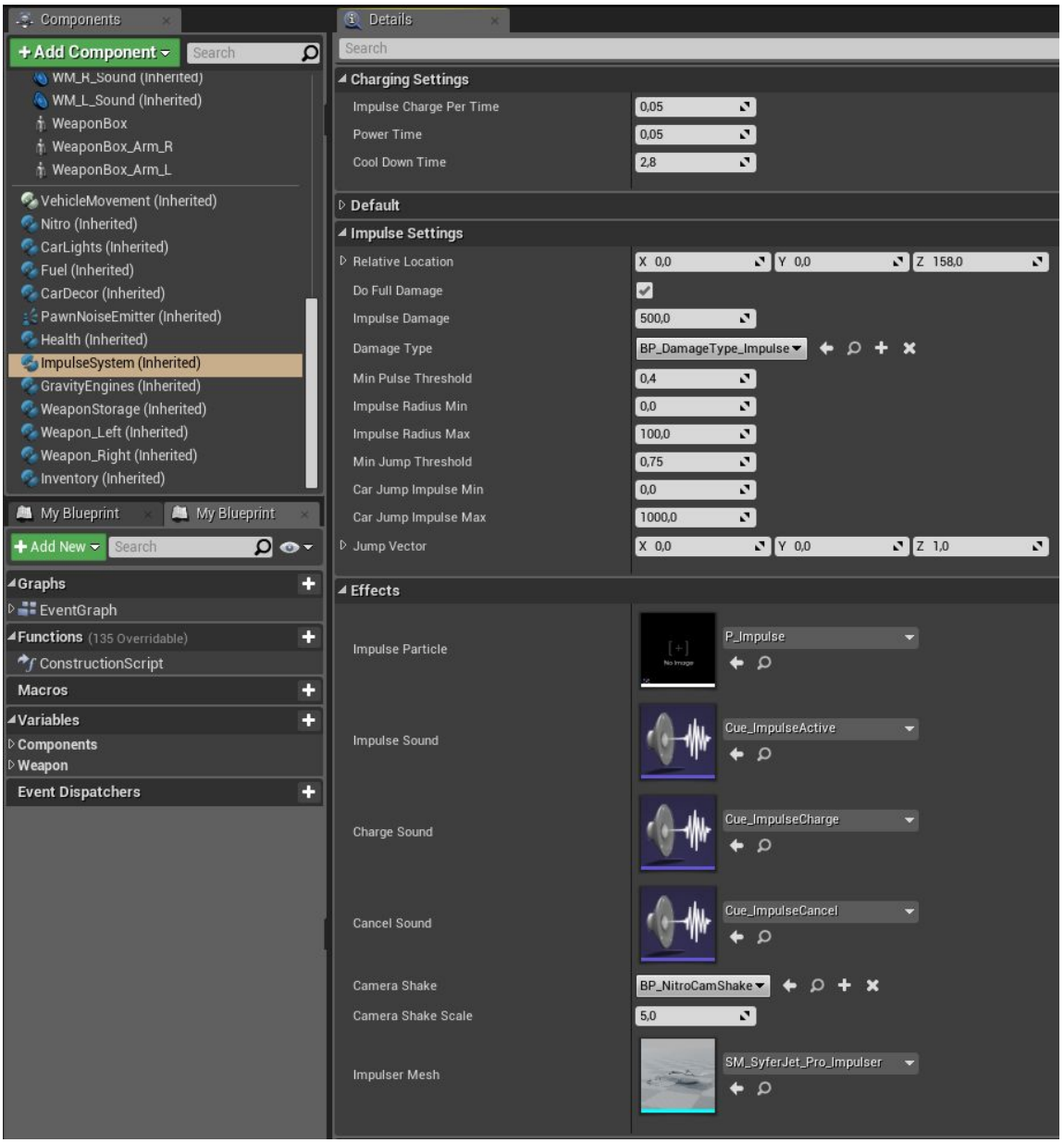
Material Index - индекс материала, на вашем автомобиле отвечающий за краску.
Dynamic Material - материал содержащий в себе карсу автомобиля.
Use Aerography - использовать аэрографию.
Aerography Texture - текстура аэрографии, по умолчанию для примера добавлены несколько аэрографий.
Base Color - цвет машины. Используется при отключенном параметре аэрографии.

Use Nitro - Данный модуль отвечает за форсированное ускорение машины. Все настройки указываются в компоненте “**Nitro**”:



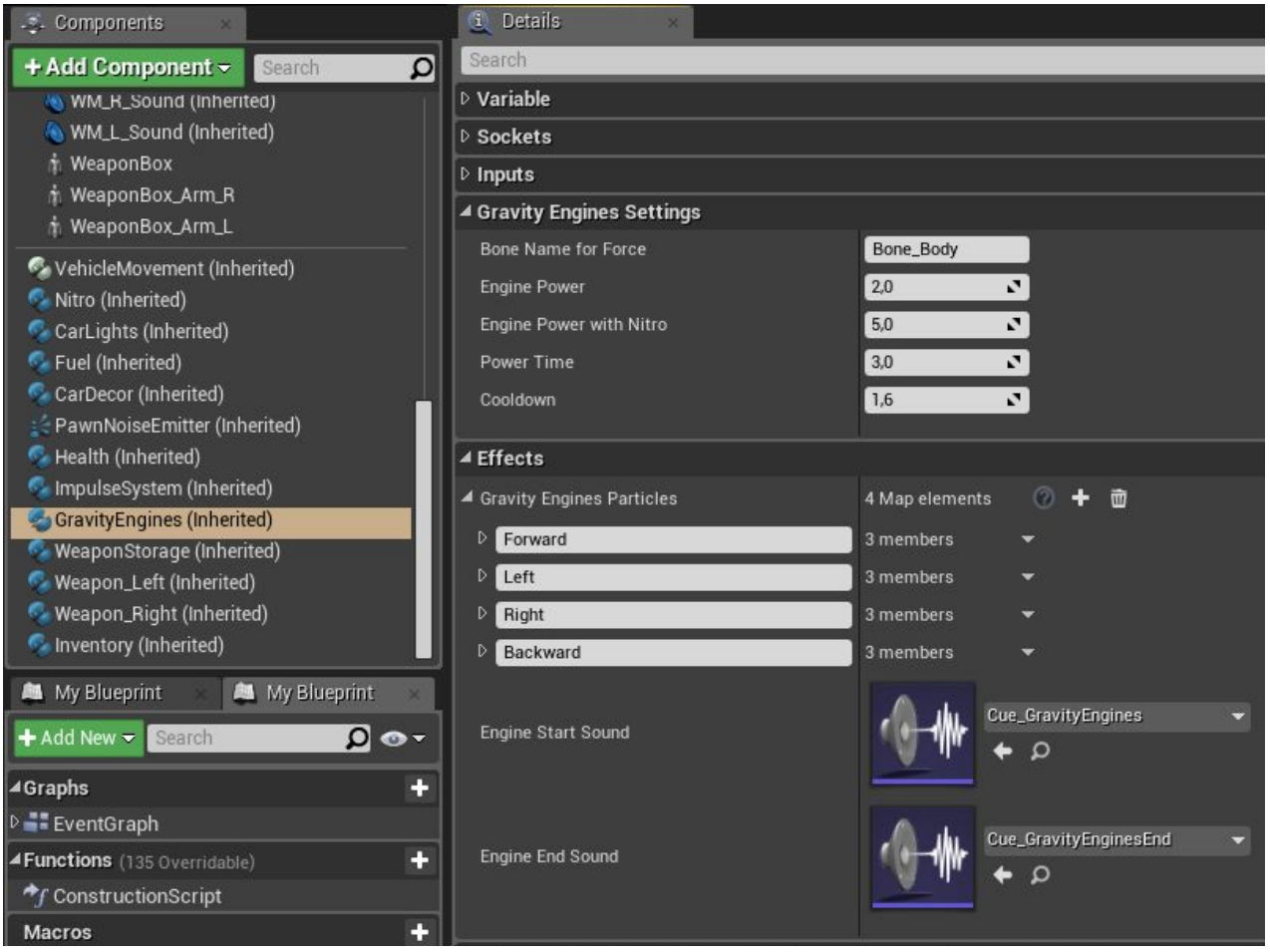
Nitro Power - Мощность ускорения.
Max Nitro - Максимальное количество нитро.
Nitro - Текущее количество нитро.
Nitro Spend Value - Кол-во траты нитро каждые 0.05 секунд.
Nitro Recovery Value - Кол-во восполняемого нитро (при 0, восполняется полностью)
Nitro Recovery Time - Время восполнения нитро.
Delay Time For Recovery Nitro - Задержка до начала восстановления.
Min Percent Nitro for Start - Минимальнео кол-во нитро для запуска ускорения.
Max Boost Speed - Максимальная ускоряемая скорость в км/ч.
В категории **Effects** настраиваются все эффекты Нитро ускорения.

Use Impulse System - Боевой модуль отвечает за импульсную установку. Настраивается в компоненте “**ImpulseSystem**”:



Charging Settings (Настройки заряда импульса):
Impusle Charge Per Time - Кол-во заряда в промежуток времени.
Power Time - промежуток времени через который заряжается импульс.
Cool Down Time - время остывания установки.
Impulse Settings (Настройки импульса):
Relative Location - Локация импульса относительно машины.
Do Full Damage - Наносить полное кол-во повреждения в радиусе.
Impulse Damage - Повреждения от импульса.
Damage Type - Тип повреждения (в классе также указывается сила физ. импульса)
Min Pulse Threshold - Кол-во процентов заряда для срабатывания импульса.
Impulse Radius min - Минимальный радиус импульса относительно заряда.
Impulse Radius Max - Максимальный радиус импульса относительно заряда.
Min Jump Threshold - Кол-во процентов заряда для срабатывания прыжка.
Car Jump Impulse Min - Минимальная сила прыжка относительно заряда.
Car Jump Impulse Max - Максимальная сила прыжка относительно заряда.
Jump Vector - Направление прыжка.
В категории “Effects” указываются все эффекты импульсной системы (включая внешний вид установки).

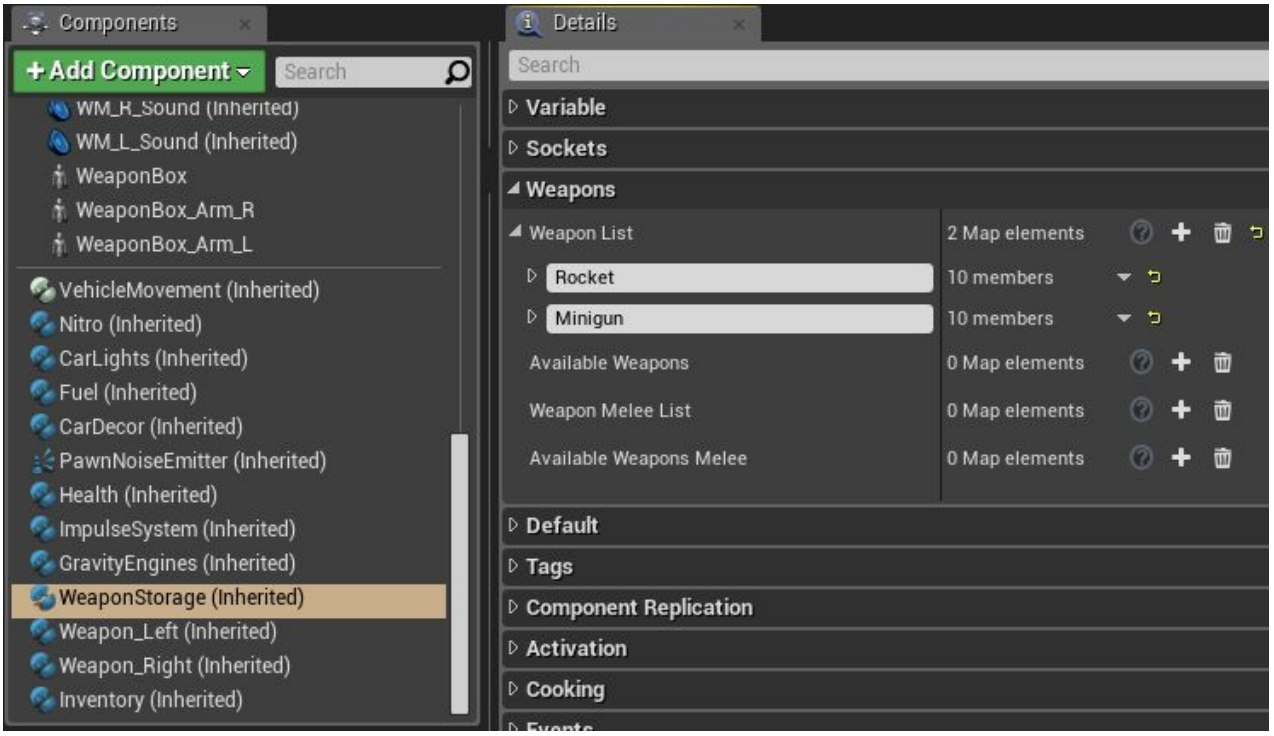
Use Gravity Engines - модуль отвечающий за гравитационные движки, которые служат для ротации машины в воздухе. По умолчанию данные движки запускаются посредством зажатия ручного тормоза в воздухе, ротацию можно усилить при активации нитро. Настройки указываются в компоненте “**Gravity Engines**”



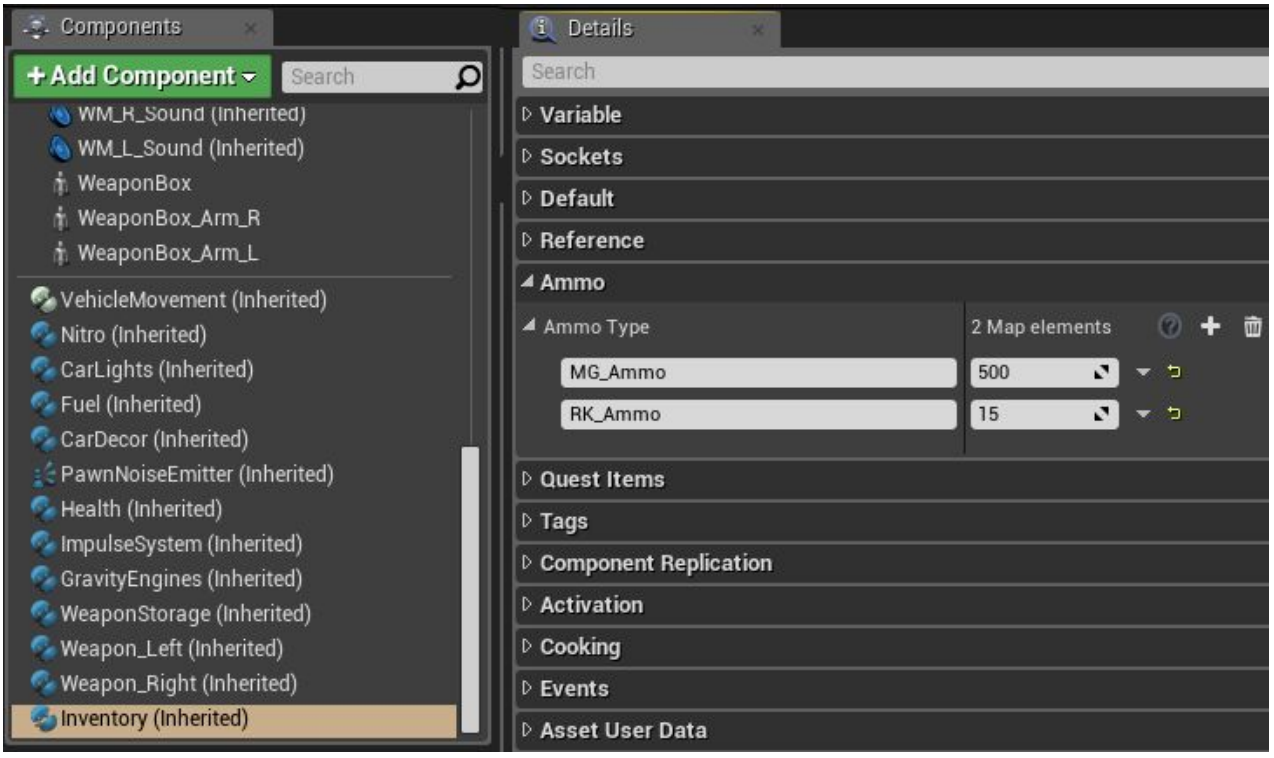
Gravity Engines Settings (базовые настройки движков):
Bone Name for Force - Название кости на которую будут влиять движки.
Engine Power - Мощность движков.
Engine Power with Nitro - Мощность движков при использовании нитро.
Power Time - Время действия движков до перегрева.
Cooldown - Время охлаждения.

В категории “*Effects*” указываются партиклы движков и их расположение,а также звуки.

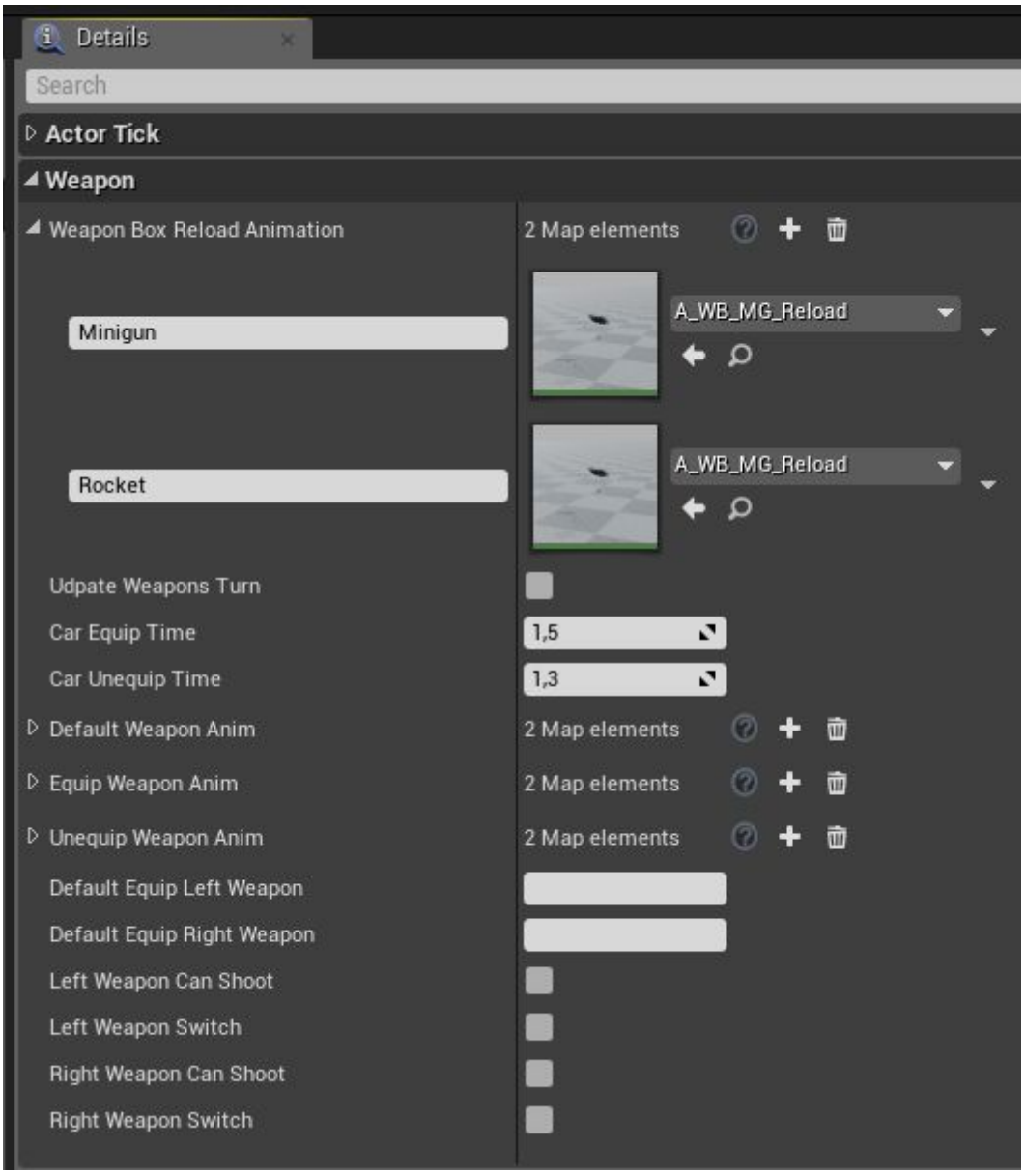
Use Weapons - Боевой модуль отвечающий за вооружение. Боевые машины по умолчанию могут использовать одновременно два оружия - слева и справа. Оружие строится на той же методике, что и оружие у персонажа, используются те же компоненты: “Weapon Storage”, “Weapon”, “Inventory” (для амуниции).
Первом делом перейдите в копоменнет “Weapon Storage” и настройте все доступное оружие для данного транспорта. По умолчанию добавлено два оружия: Миниган и ракетница.



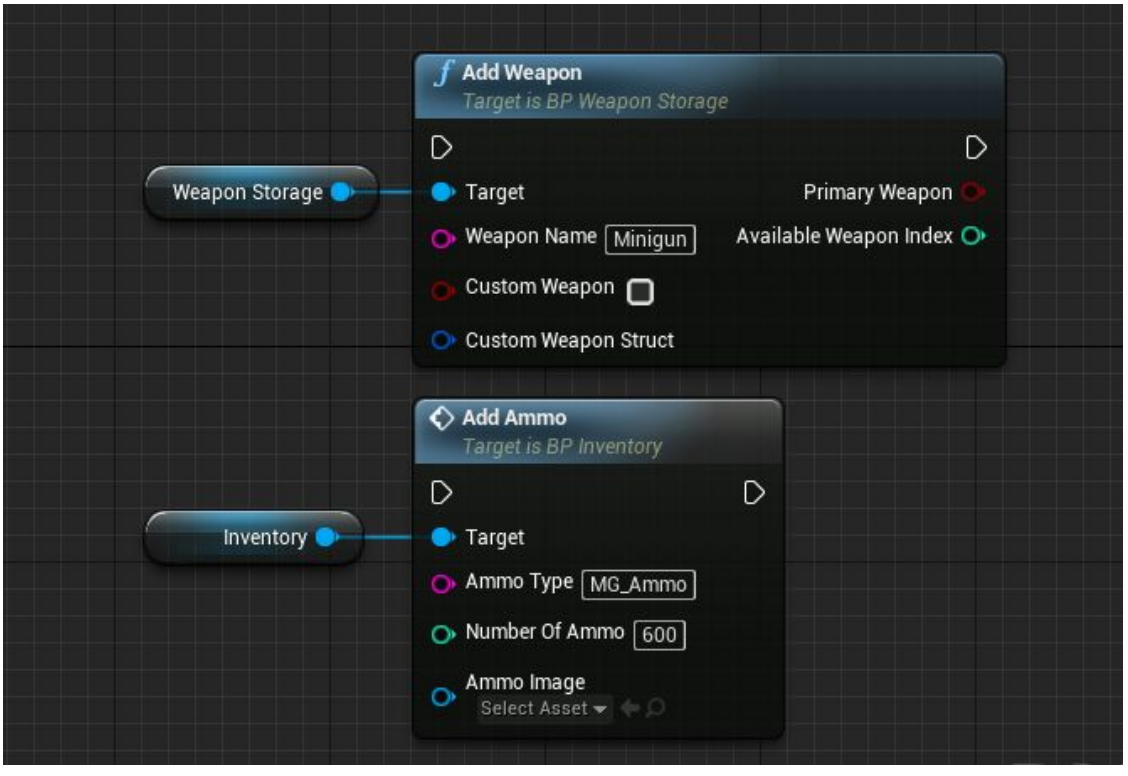
После добавления оружия добавьте необходимые патроны в компоненте Inventory:



Следующим шагом будет добавление непосредственно на машину созданного оружия. Это можно сделать указав названия оружий по умолчанию в базовых настройках в категории Weapon, названия параметров *Default Equip Left Weapon* и *Default Equip Right Weapon.* Также в данной категории можно указать дополнительные анимации, такие как анимации перезарядок для гибкой подводки обоймы и прочие:

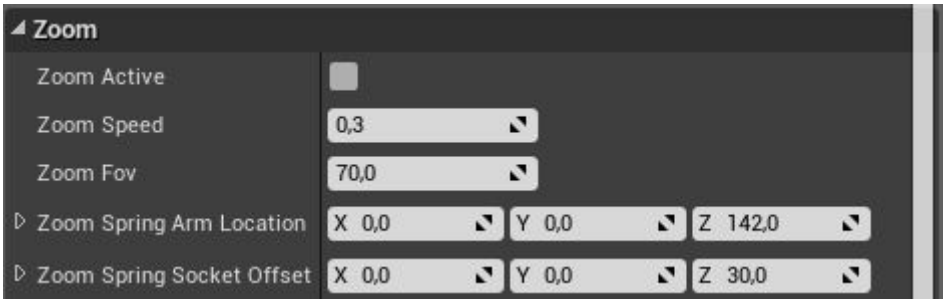


Второй способ добавления оружия и патрон. Используете следующие узлы для добавления оружия и патрон:



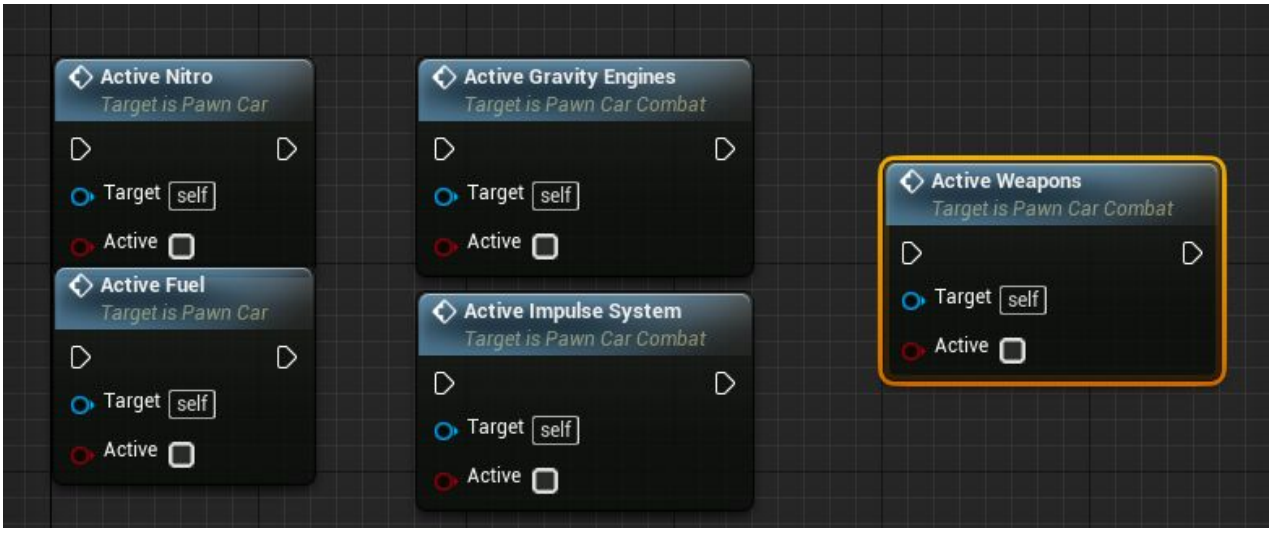
Внимание машина использует все добавленное оружие и переключается между ними посредством кнопок 1 и 2. Другими словами если вы хотите установить два минигана на машину вам нужно добавить два оружия с именем Minigun. По умолчанию чтобы снять оружие зажмите и удерживаете 1 или 2.

Вооружение машины также поддерживает параметры приближения (Zooming). Активируется при выдвинутом оружии.
Перейдите в базовые настройки в категорию Zoom:



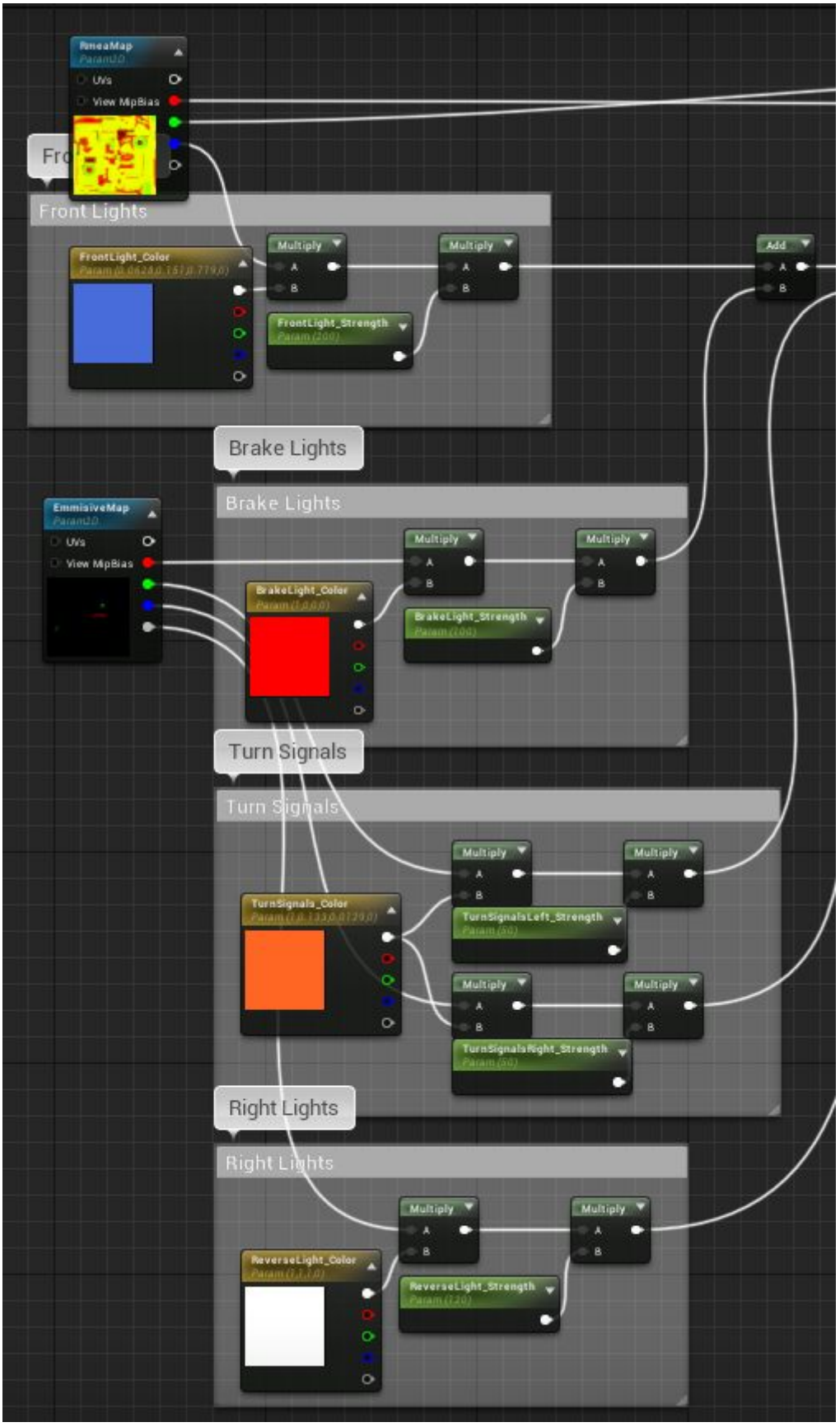
Zoom Speed - Скорость приблежения в секндах.
Zoom Fov - Поле зрения при приближении.
Zoom Spring Arm Location - Положение spring arm при приближении.
Zoom Spring Socket Offset - Положение камеры относительно спринг арм при приближении.

Активировать/Деактивировать модули в реальном времени вы можете при помощи функций в категории “CarModules”:



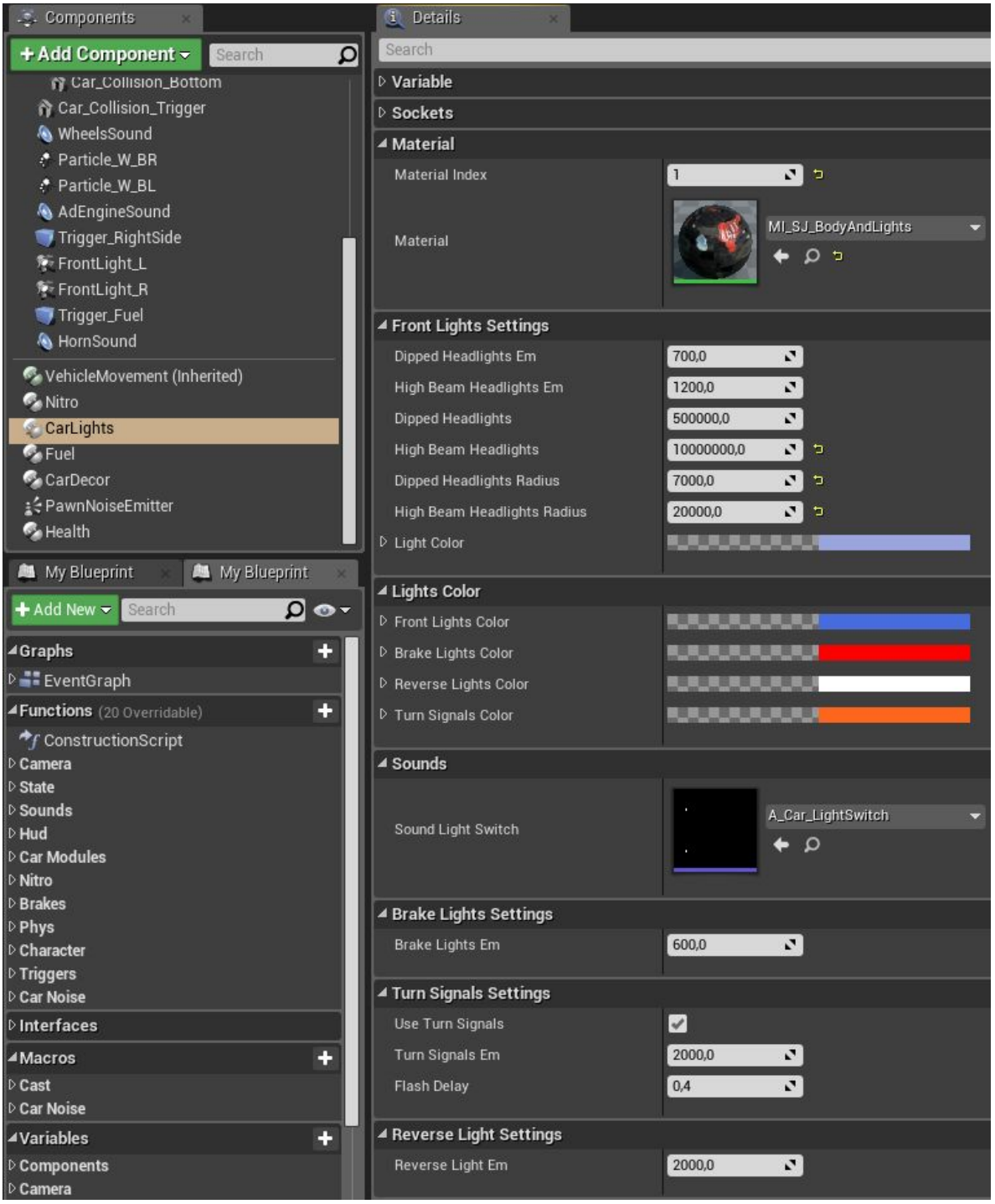
Компоннет **CarLights.** Данный компонент служит для настройки фар машины и сигналов (задний ход, поворотники, тормоз). Первым делом нужно подготовить текстуры (маски) для вашей машины. Маски для фар должны располагаться в следующих текстурах и каналах:
1. Маска для передних фар должна находиться в синем канале текстуры масок (RMEA).
Для остальных сигналов предназначена отдельная текстура.(по умолчанию это T_Corpus_Emissive_M).
2. Маска для сигнала тормоза должна располагаться в красном канале.
3. Маска для сигнала левого поворотника должна располагаться в зеленом канале.
4. Маска для сигнала правого поворотника должна располагаться в синем канале.
5. Маска для сигнала заднего хода должна располагаться в альфа канале.

Вы можете открыть материал “**M_SJ_Body**” и посмотреть там каналы:



Передние фары машины поддерживают 3 режима света:
Выключены
Ближний свет
Дальний свет

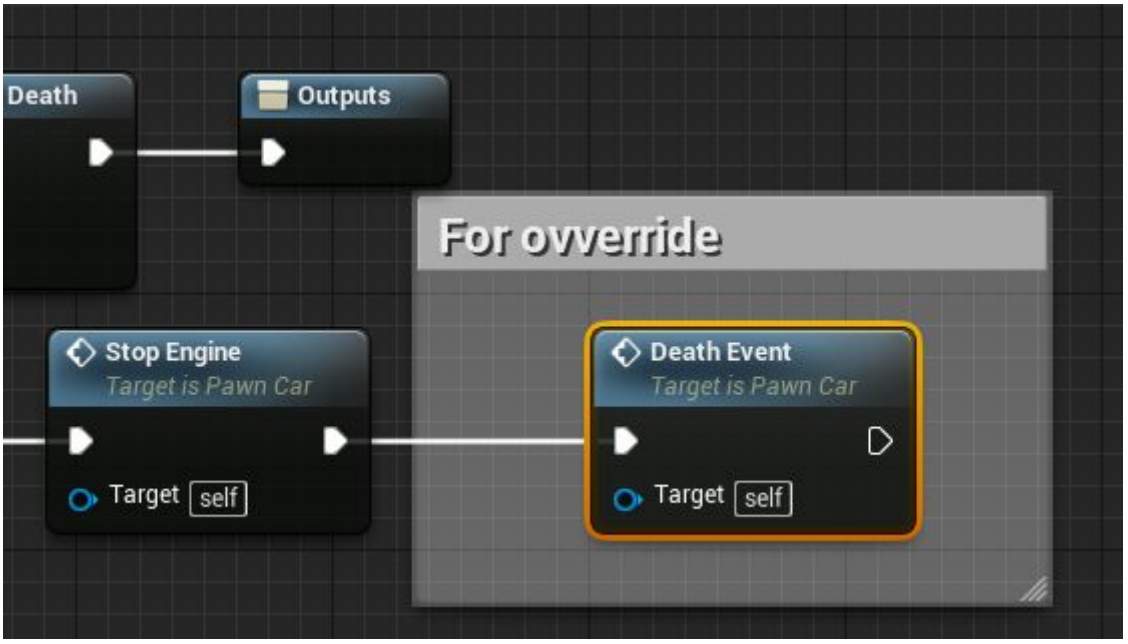
Перейдем к настройкам сигналов машины. Выделите компонент “Car lights”.



В категории “**Material**” нужно выбрать ваш материал и его индекс на модели.
В категории “**Front Lights Settings**” настраивается свет передних фар.
Dipped Headlights Em - Сила свечения (глоу) от фар при ближнем свете.
High Beam Headlights Em - Сила свечения (глоу) от фар при дальнем свете.
Dipped Headlights - Яркость света от фар при ближнем свете.
High Beam Headlights - Яркость света от фар при дальнем свете.
Dipped Headlights Radius - Дальность освещения при ближнем свете.
High Beam Headlights Radius - Дальность освещения при дальнем свете.
Light Color - Цвет освещения от передних фар.

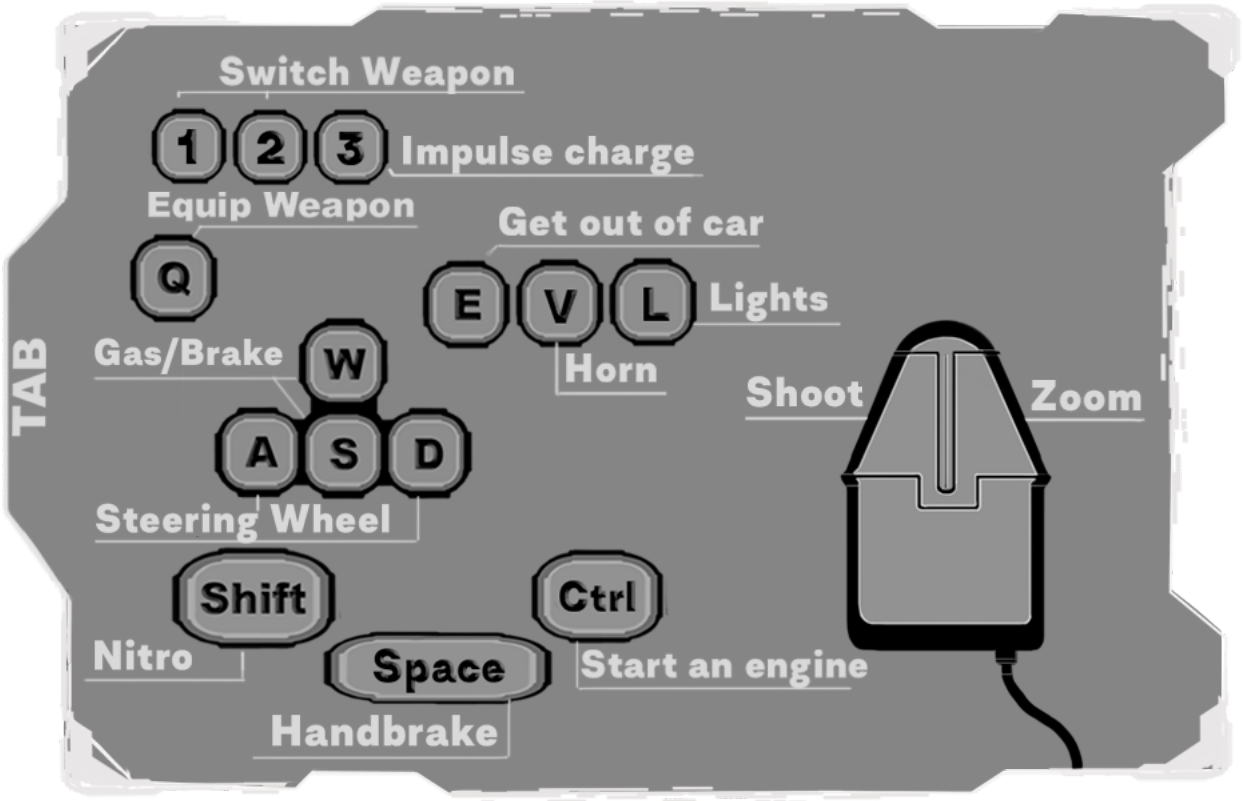
В категории “**Lights Color**” настраивается свет остальных сигналов.
В категории “**Sounds**” выбирается звук переключения режимов передних фар.
В категории “**Brake Lights Em**” настраивается сила свечения сигнала тормоза.
В категории “**Turn Signals Settings**” настраиваются сигналы поворотников.

В категории “**ReverseLightSett**” настраивается сила свечения сигнала заднего хода.
Так же машина содержит в себе компонент здоровья “**Health**” по умолчанию при отсутствии здоровья машина просто глохнет. Добавлена специальная функция для перезаписи “**DeathEvent**” - перезаписывая данную функцию вы сможете добавить любые другие ивенты для смерти машины к примеру взрыв.



Также в машине предусмотрена клавиша зажигания (ctrl). Может пригодится при экономии бензина или если нужно, чтобы машина не издавала звуков для привлечения врагов.

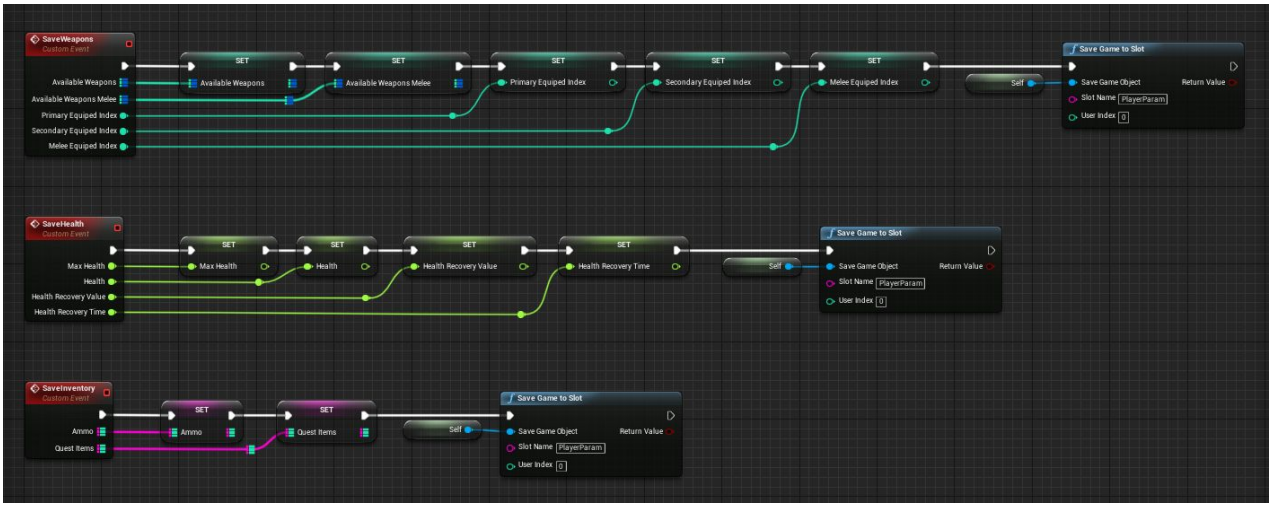
Базовое управление автомобилем:



Сохранение и загрузка игры.

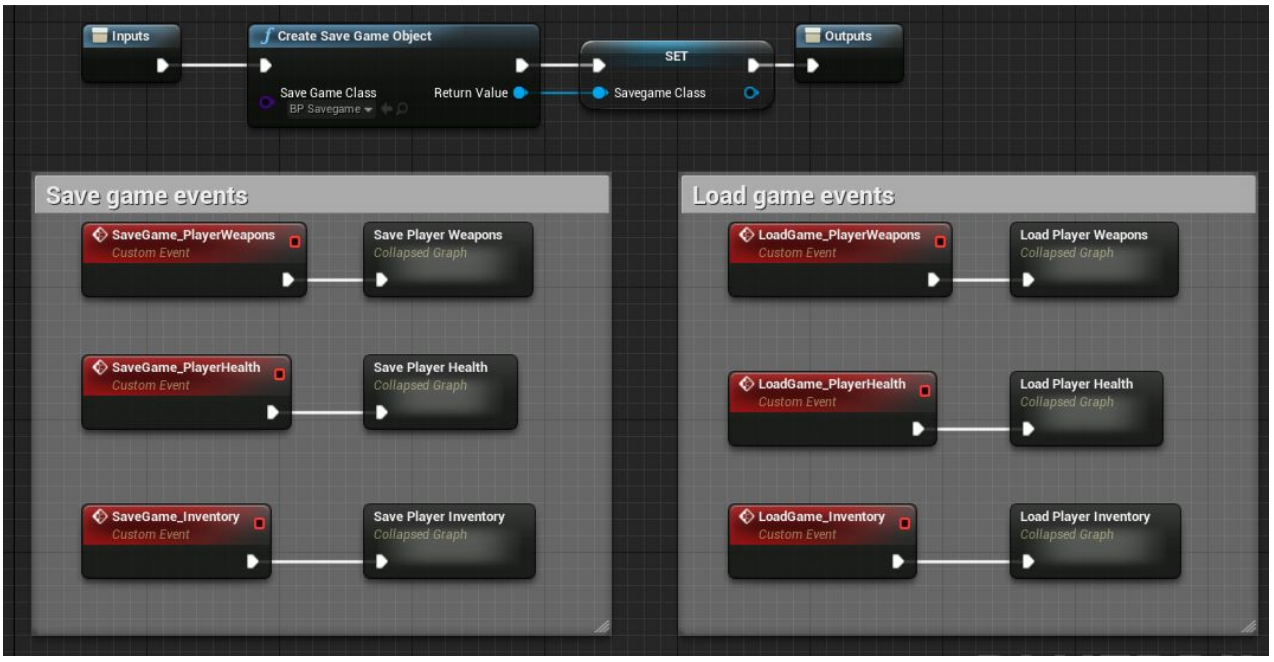
Так как было много вопросов по поводу сохранения персонажа. В проект добавлен класс сохранения игры, который сохраняет оружие, инвентарь и здоровье персонажа.

Данный класс служит для сохранения переменных более подробно тут - SaveGame. Данный класс использует слот сохранения “PlayerParam”, в данном слоте сохраняются параметры игрока.



В павне игрока есть ивенты в блоке “**SaveGameComp**”. Ивенты сохранения обращаются к классу “**BP_Savegame**” и сохраняют текущие параметры игрока. Ивенты загрузки получают сохраненные данные из класса “**BP_Savegame**” и применяют их к персонажу. На примере оружия - обнуляется все оружие и добавляются все сохраненное оружие и данных в “**BP_Savegame**”.

Для теста в данном блоке добавлены инпуты 6 и 7: 6 - сохранение, 7 - загрузка.



При помощи данного класса вы можете расширить систему сохранения к примеру сохранив позицию игрока, окружающих врагов и прочее.. Чтобы было проще ориентироваться во всем коде присутствуют большое кол-во комментариев.

Онлайн канал поддержки - [Discord channel].
Спасибо за внимание.
С наилучшими пожеланиями, Андрей (ZzGERTzZ).