ЛЕКЦИЯ 3

Создание и использование Актор Классов

ЦЕЛИ И ИТОГИ ЛЕКЦИИ

Goals

Цели этой лекции

- Показать, как добавлять компоненты в Blueprint
- Знакомство со строительным скриптом (Construction Script)
- Показать различные типы событий
- Показать как создавать, уничтожать и ссылаться на экземпляры Actor.
- Показать, как выполнять сопоставления ввода

Outcomes

К концу этой лекции вы сможете

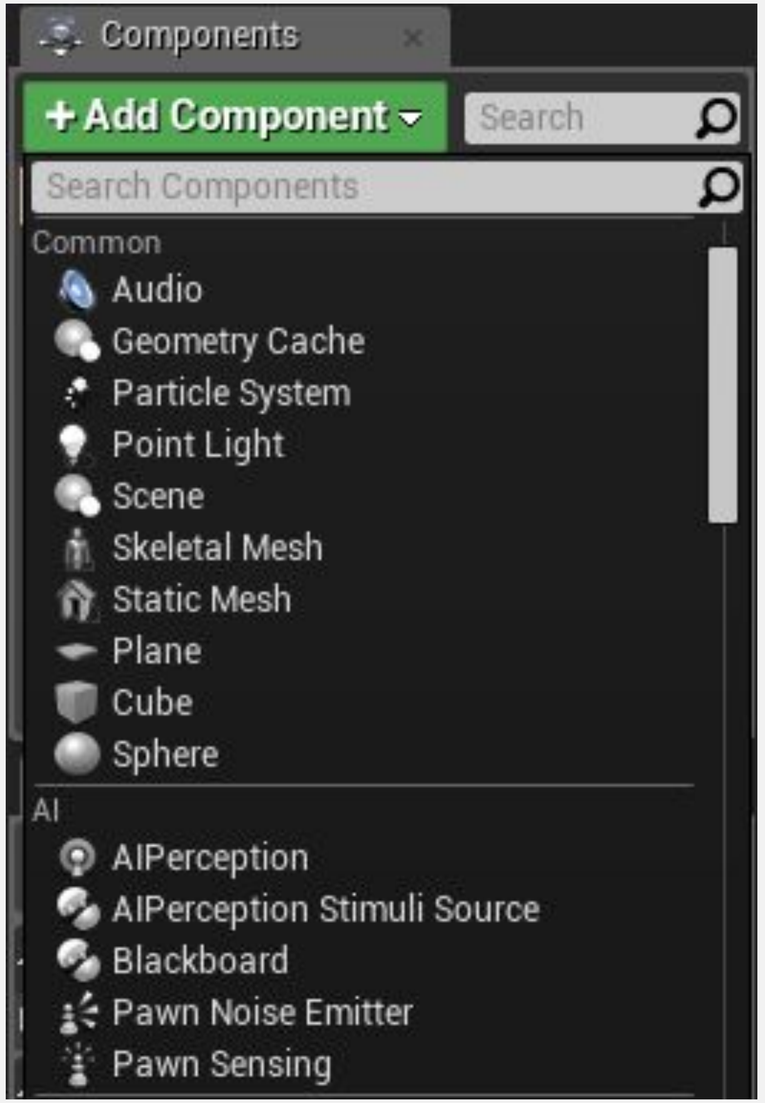
- Добавить различные типы компонентов в Blueprint
- Управление экземплярами Actor
- Создание событий коллизии и мыши
- Создание событий ввода, а также ключей и осей карты

КОМПОНЕНТЫ

Компоненты - это готовые к использованию классы, которые можно использовать внутри Blueprints. Некоторые функции могут быть включены в Blueprint, использующие только компоненты.

Чтобы добавить компоненты в Blueprint, используйте панель **Components** в редакторе **Blueprint**.

На изображении справа показана панель **Components** для нового Blueprint с некоторыми параметрами компонентов, которые отображаются при нажатии кнопки **Add Component**.

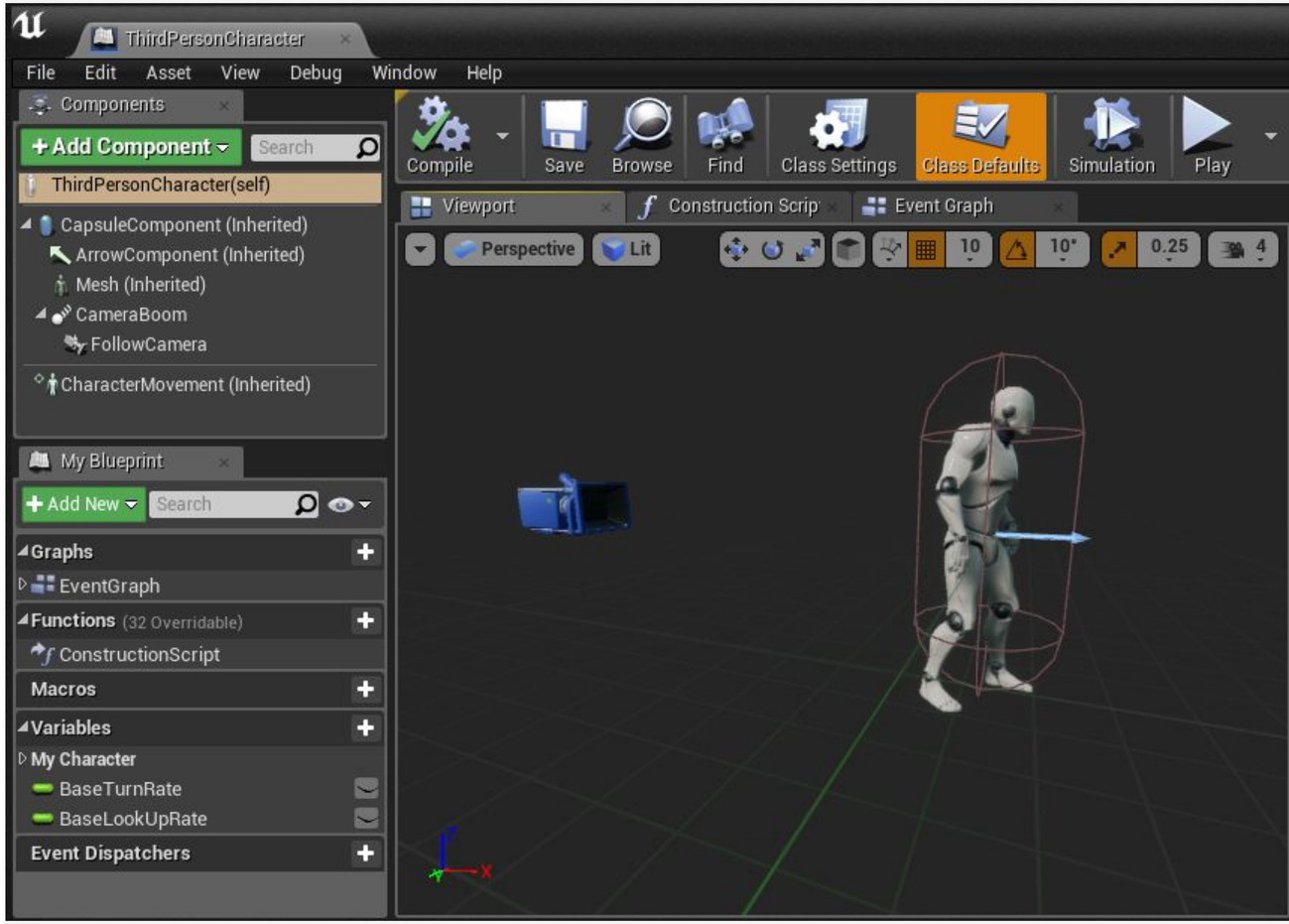


КОМПОНЕНТЫ: ВЬЮПОРТ

Визуальное представление компонентов можно увидеть во **Вьюпорте**.

На изображении справа показаны компоненты, которые являются частью Блюпринта **ThirdPersonCharacter** из шаблона от третьего лица. Компоненты, у которых есть «(**Inherited**)» рядом с именем, были унаследованы от класса **Character**.

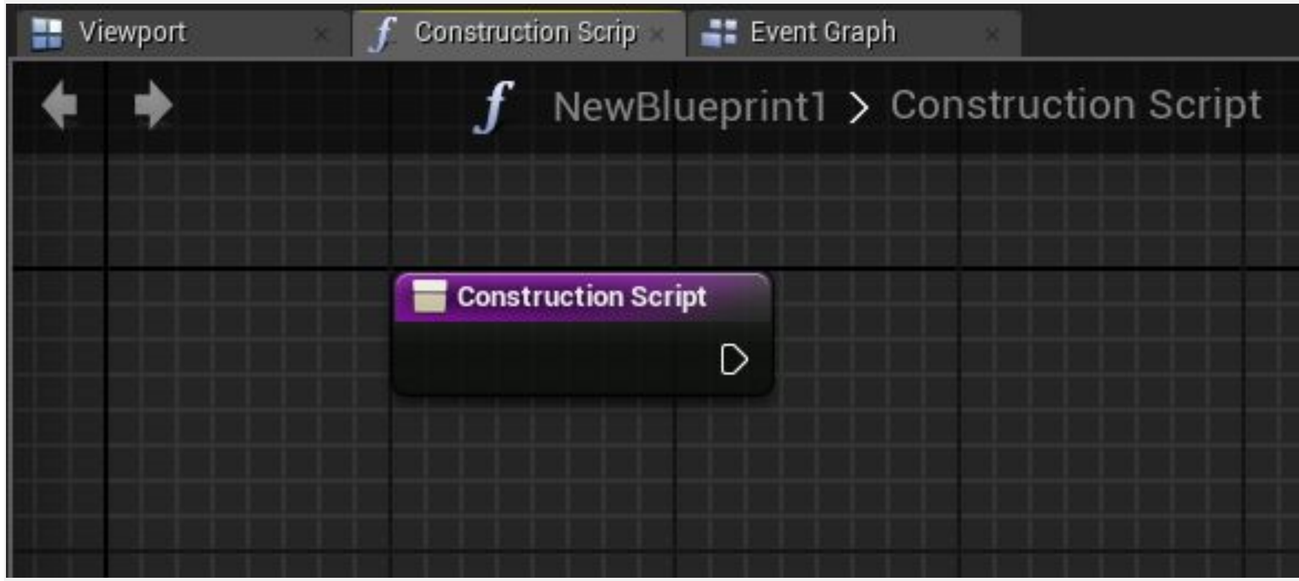
- **CapsuleComponent** используется для тестирования коллизии.
- Компонент **Mesh** - это скелетный меш, который визуально представляет персонажа.
- Компонент **FollowCamera** - это камера, которая будет использоваться для просмотра игры..
- Компонент **CharacterMovement** содержит различные свойства, которые используются для определения движения.



CONSTRUCTION SCRIPT

Construction Script - это специальная функция, которую выполняют все Блюпринты Акторов при первом добавлении Блюпринта на уровень, при изменении его свойств или при спавне класса во время выполнения. В Construction Script есть отдельный график, на котором можно разместить выполняемые действия.

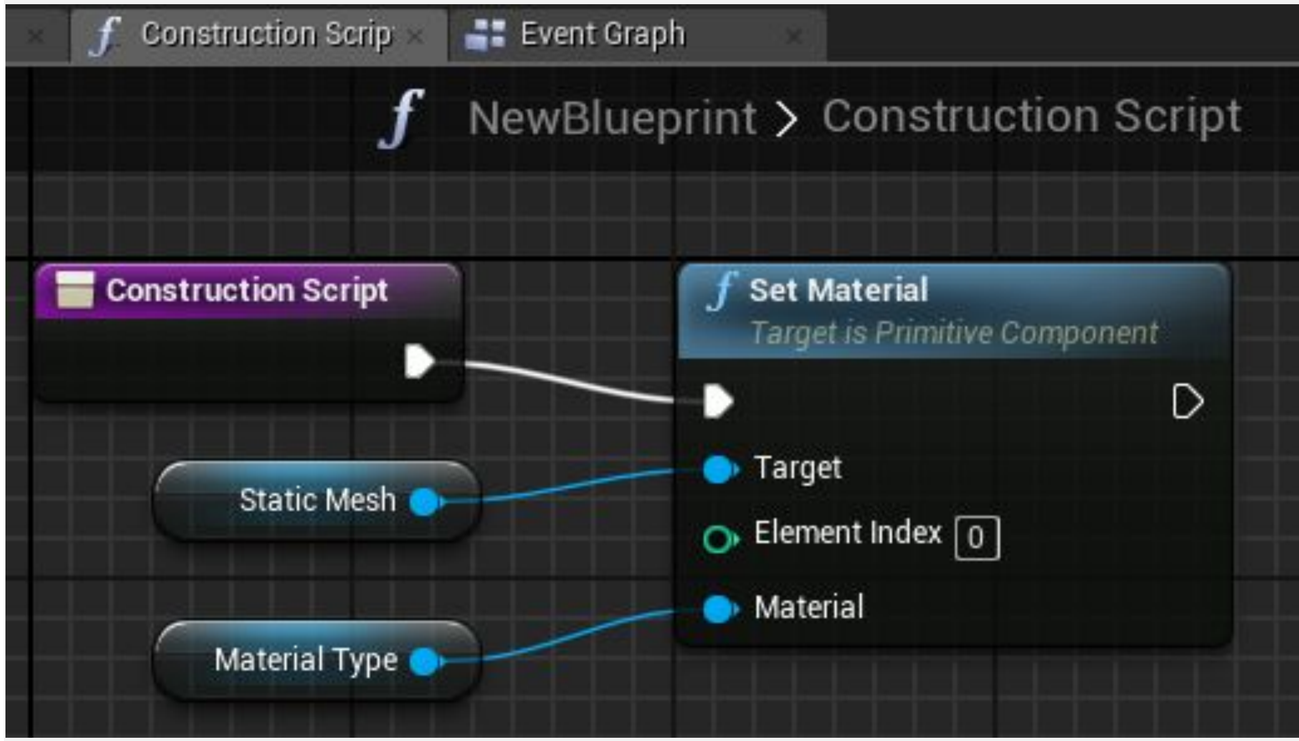
Важно отметить, что Construction Script не запускается на размещенных Акторах при запуске игры.



CONSTRUCTION SCRIPT: ПРИМЕР

Construction Script, показанный справа, использует функцию **Set Material** для определения типа материала компонента Стакик Меша в соответствии с материалом, выбранным в редактируемой переменной **Material Type**.

Каждый раз, когда изменяется переменная **Material Type**, **Construction Script** запускается снова, обновляя объект новым материалом.

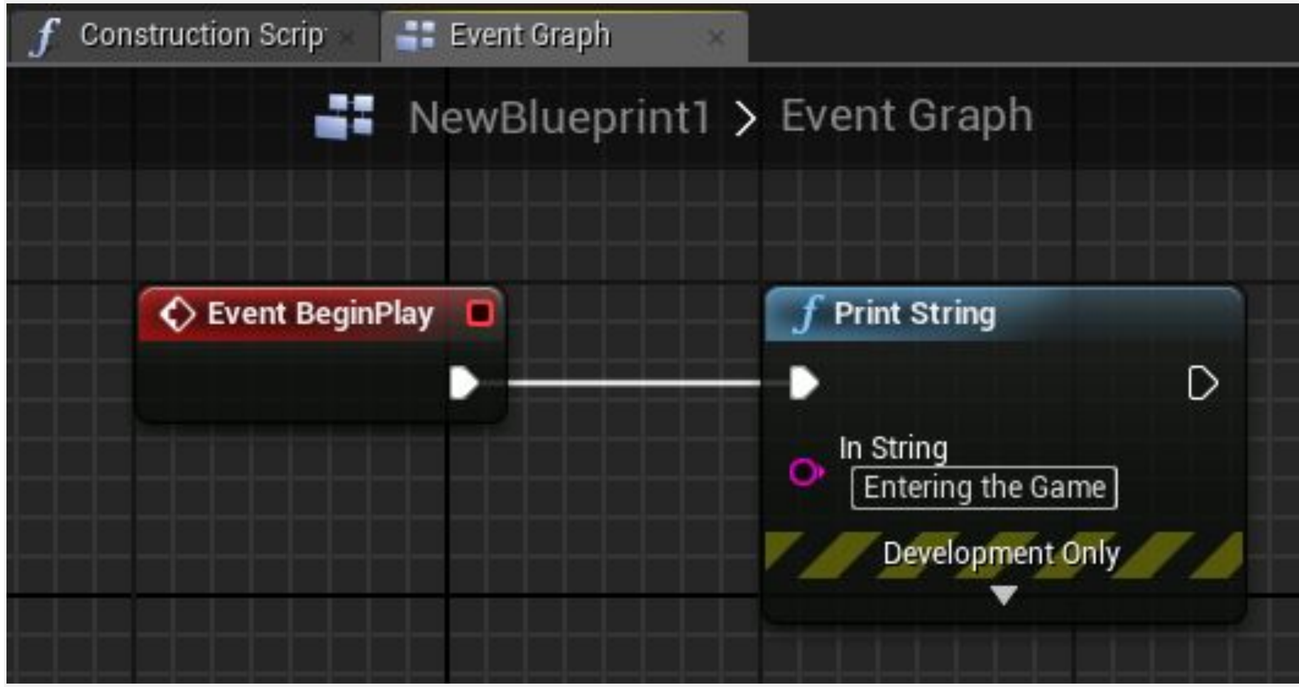


СОБЫТИЯ

СОБЫТИЕ BEGIN PLAY

События позволяют взаимодействовать Unreal Engine с Акторами.

Типичный пример - событие **BeginPlay.** Событие **BeginPlay** запускается при запуске игры для Актора. Если Актор появляется в середине игры, это событие запускается немедленно.



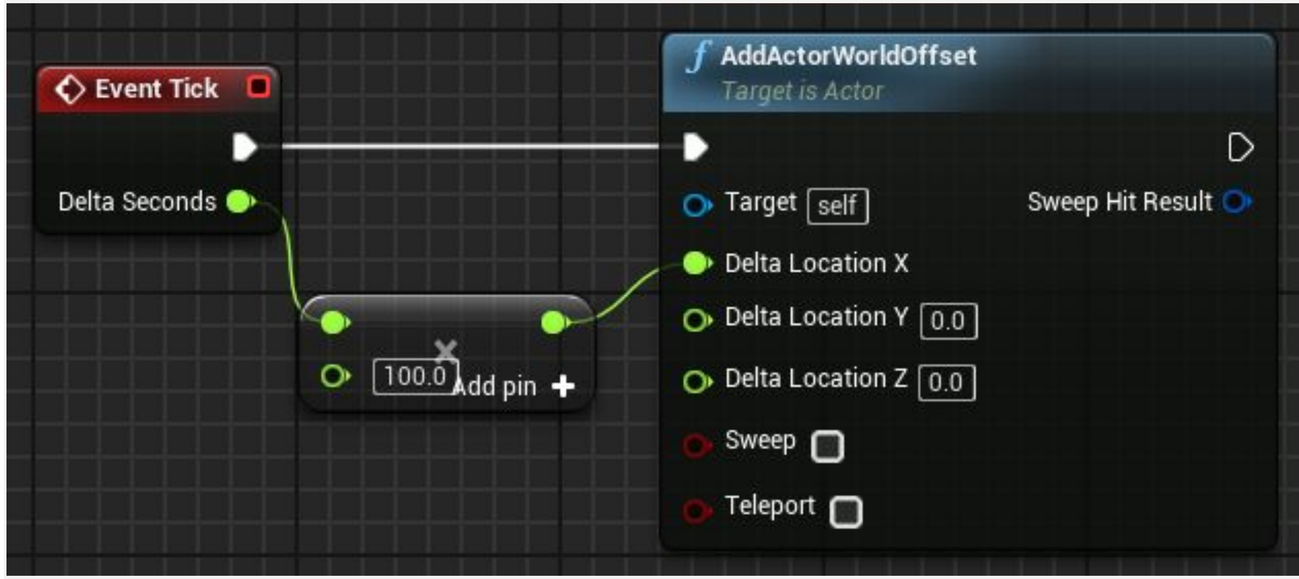
СОБЫТИЕ TICK

Есть событие под названием «**Tick**», которое вызывается в каждом кадре игры. Например, в игре, которая работает со скоростью 60 кадров в секунду, событие **Tick** вызывается 60 раз в секунду.

Событие **Tick** имеет параметр, известный как **Delta Seconds**, который содержит количество времени, прошедшее с момента последнего кадра.

В событии **Tick**, показанном справа, Актор движется по оси X со скоростью 100 сантиметров в секунду.

Используйте событие Tick только при необходимости, так как это может повлиять на производительность.



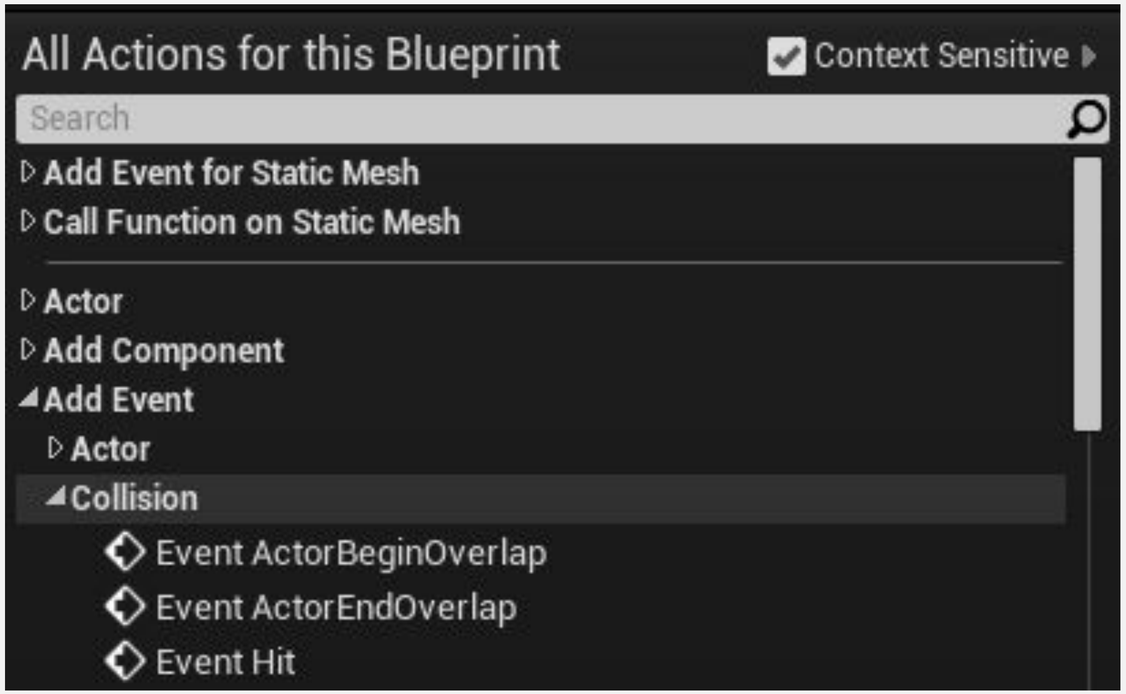
СОБЫТИЯ COLLISION

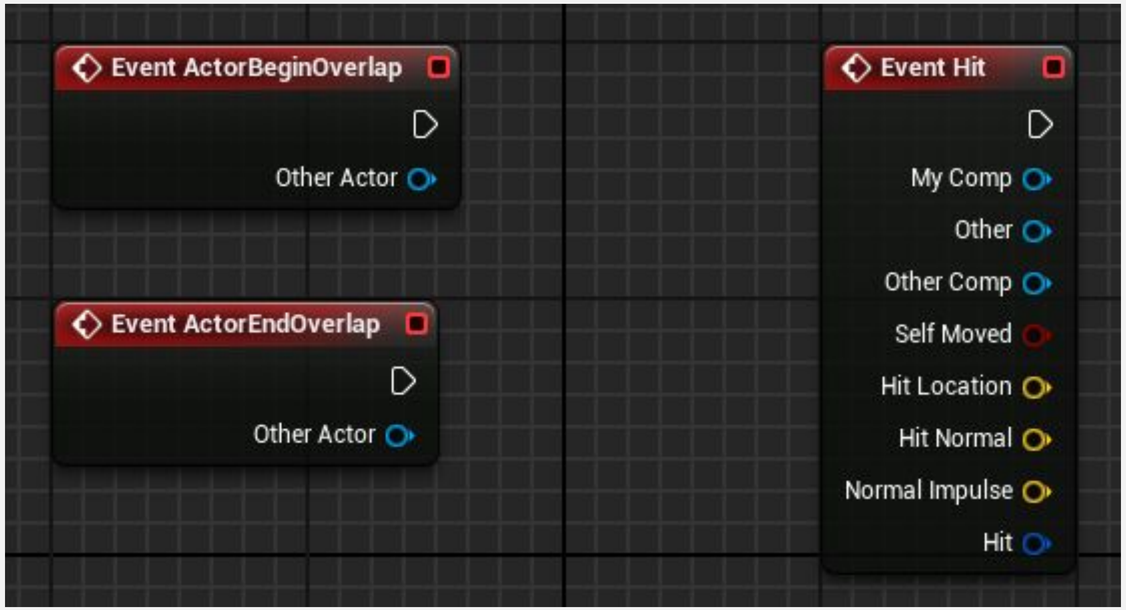
События столкновения запускаются, когда два Актора сталкиваются или перекрываются.

Событие **ActorBeginOverlap** будет выполняться, когда два Актора начинают перекрываться, а для свойства **Generate Overlap Events** обоих Акторов установлено значение **«true».**

Событие **ActorEndOverlap** будет выполняться, когда два Актора перестанут перекрываться**.**

Событие Hit будет выполнено, если для свойства **Simulation Generates Hit Events** одного из Акторов в столкновении установлено значение **«true».**





ВЗАИМОДЕЙСТВИЯ МЫШИ

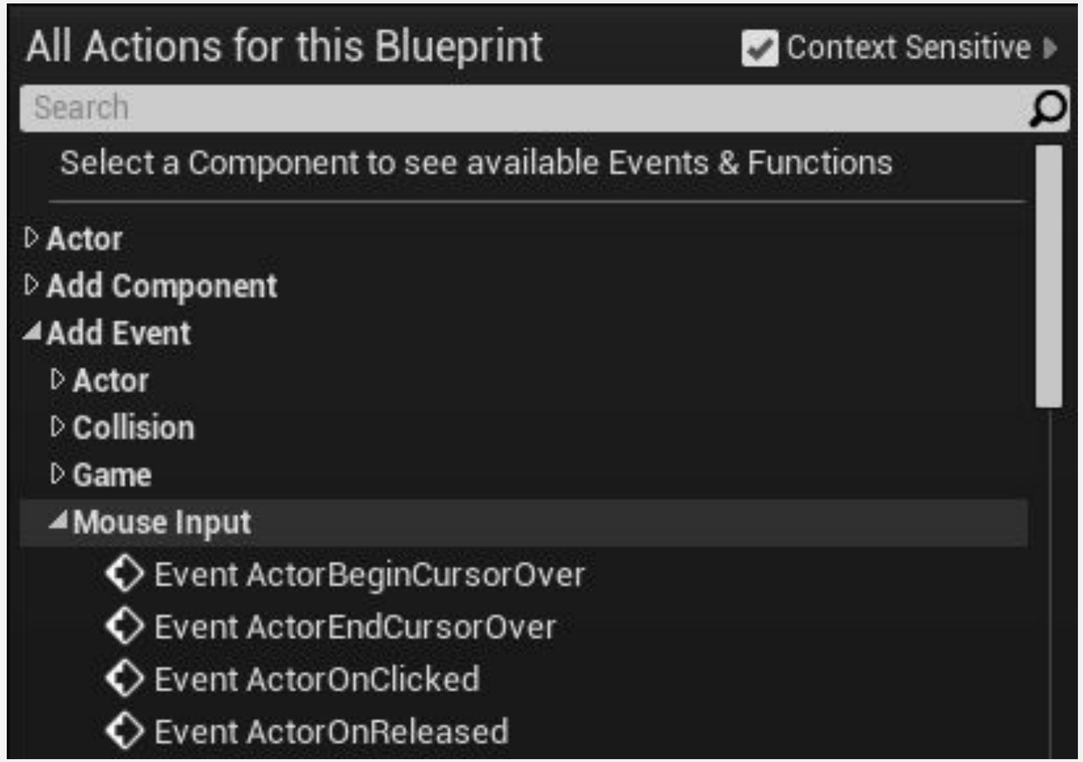
Есть некоторые события, связанные с **взаимодействием мыши** с Актором.

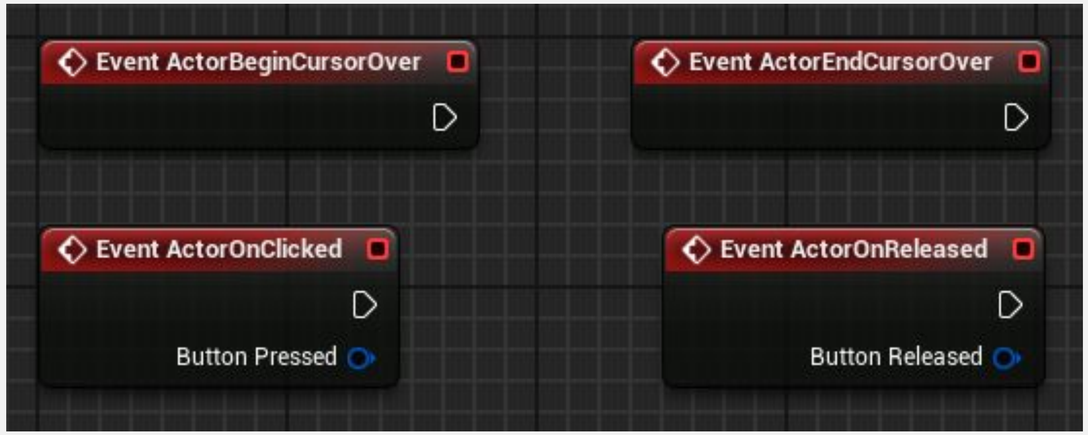
Событие ActorBeginCursorOver вызывается, когда курсор мыши перемещается по Актору.

Событие ActorEndCursorOver вызывается, когда курсор мыши уходит с Актора.

Событие ActorOnClicked вызывается, когда мышь щелкает по Актору.

Событие ActorOnReleased вызывается, когда кнопка мыши отпускается, а курсор мыши все еще находится над Актором.





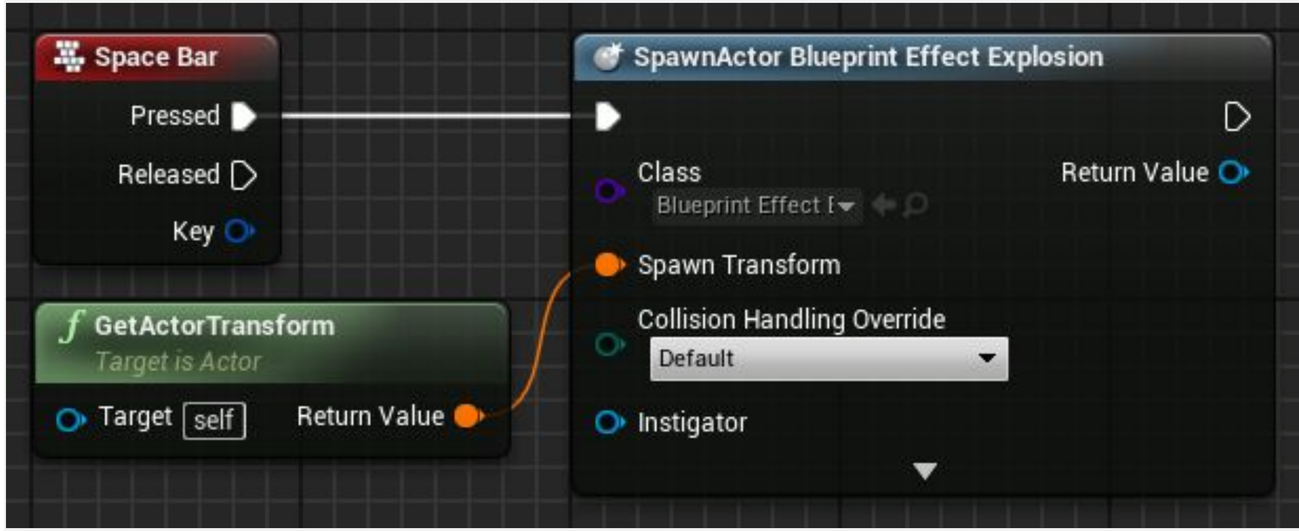
ЭКЗЕМПЛЯРЫ АКТОРОВ

СПАВН АКТОРОВ

Spawn Actor from Class - это функция, которая создает экземпляр Актора, используя указанный класс и преобразование.

Входные данные **Collision Handling Override** определяют, как обрабатывать столкновение во время создания. Выходной параметр **Return Value** - это ссылка на вновь созданный экземпляр.

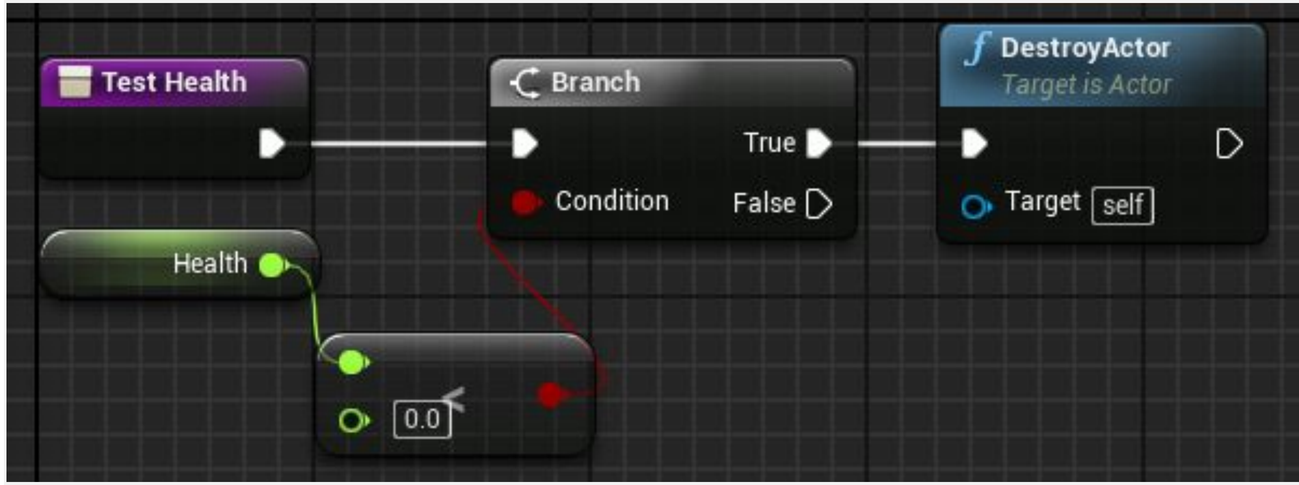
В примере справа, когда нажата **клавиша пробела**, создается экземпляр класса **Blueprint Effect Explosion** в том же месте (преобразовании) текущего Blueprint.



УНИЧТОЖЕНИЕ АКТОРОВ

Функция **DestroyActor** удаляет экземпляр Актора из уровня во время выполнения. Удаляемый экземпляр должен быть указан в параметре **Target**.

На изображении справа показана функция с именем «**Test Health**», которая проверяет, меньше ли значение переменной **Health**, чем ноль. Если «**true**», текущий экземпляр этого Blueprint, который представлен «**self**», будет уничтожен.



GET ALL ACTORS OF CLASS

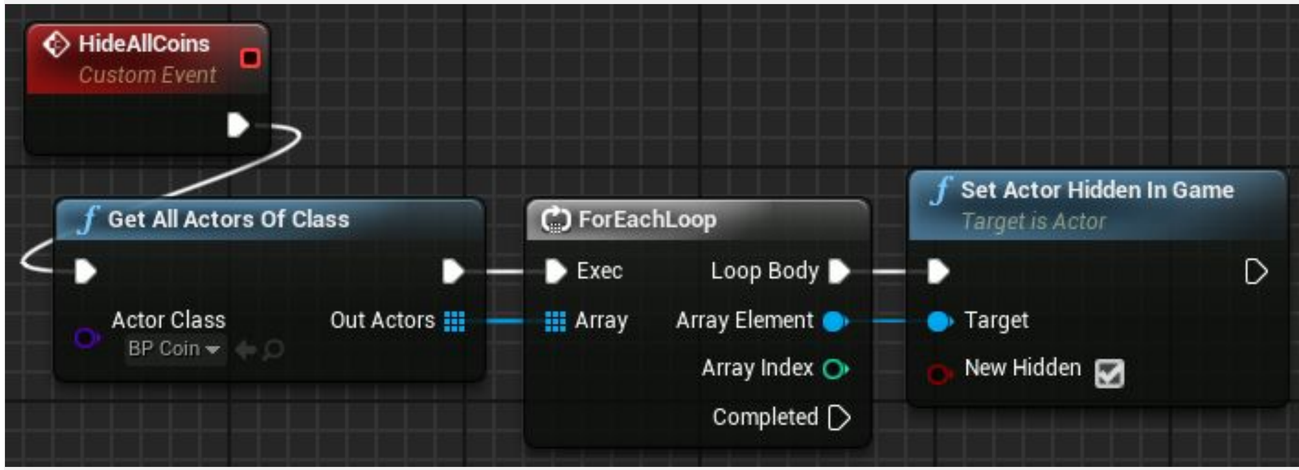
Get All Actors Of Class - это функция, которая получает ссылки на всех Акторов на текущем уровне, которые принадлежат указанному классу.

Параметр **Actor Class** указывает класс, который будет использоваться при поиске.

Выходной параметр **Out Actors** - это массив, содержащий ссылки на экземпляры Акторов указанного класса, найденные на уровне.

На изображении справа **Get All Actors Of Class** возвращает массив Акторов **BP_Coin**, затем использует ноду **ForEachLoop** для установки свойства **New Hidden** в значение «**true**» в функции **Set Actor Hidden In Game** для каждого Актора в массиве.

Это может быть дорогостоящей операцией. Не используйте его в событии Tick.



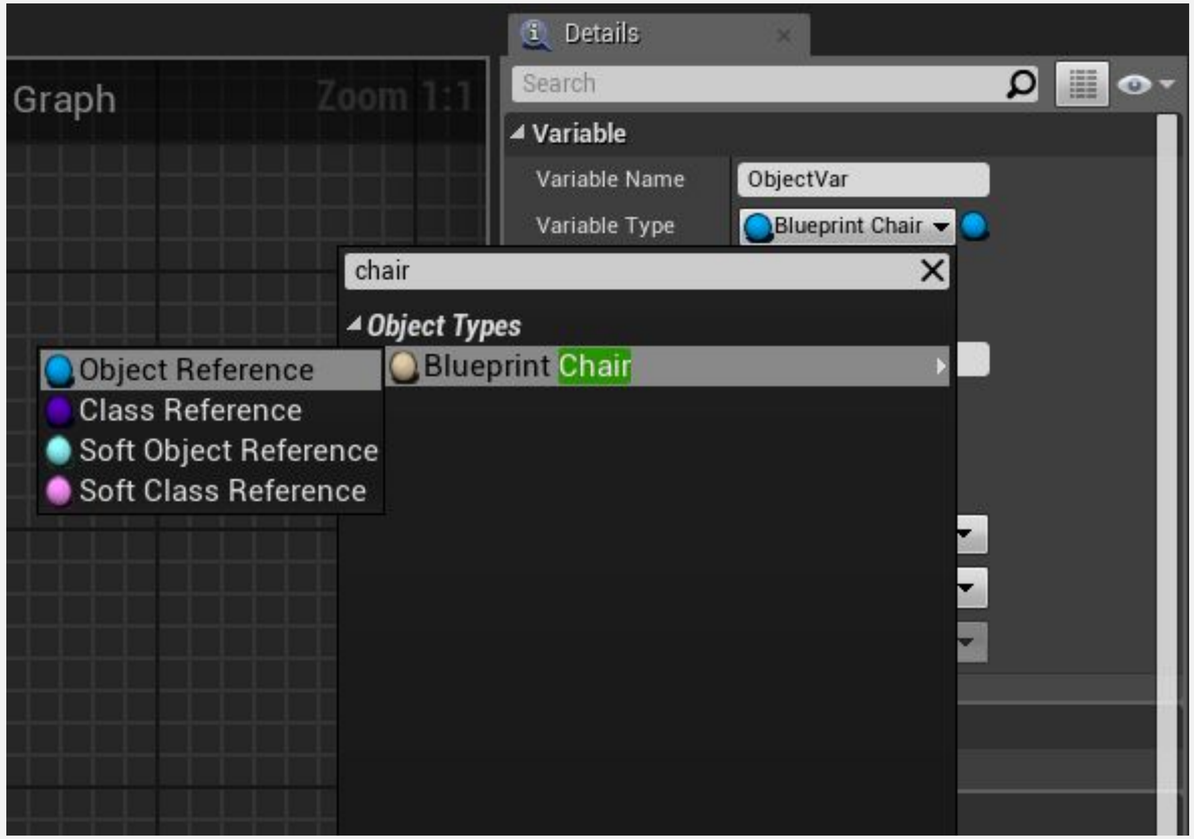
ССЫЛКИ НА АКТОРЫ

В Blueprint можно создать переменную, которая ссылается на объект / актор.

Пример справа показывает создание переменной, которая ссылается на экземпляр класса **Blueprint_Chair**. Ссылка на объект указывает на Актора, находящегося на Уровне.

При создании переменная пуста. Способ установить экземпляр для этой переменной - проверить свойство **Instance Editable**, добавить Blueprint на уровень и на панели **Details** выбрать Актора, который находится на уровне.

Другой способ - использовать возвращаемое значение функции **Spawn Actor from Class**.



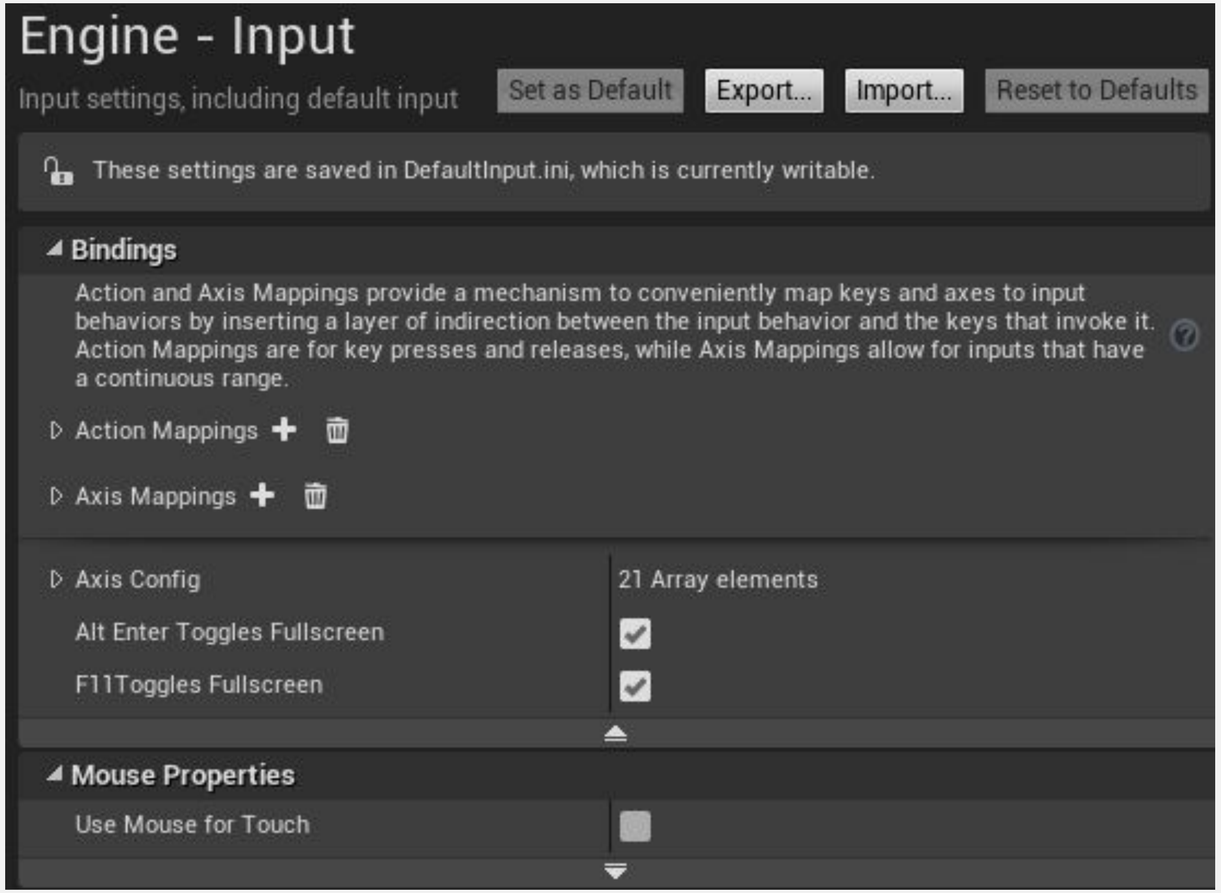
ВВОД ИГРОКА

ВХОДНЫЕ СОПОСТАВЛЕНИЯ

Можно создавать новые входные события, которые представляют действия, которые имеют смысл в игре.

Например, вместо создания события ввода для ЛКМ, которое запускает пистолет, лучше создать событие действия под названием «**Fire**» и сопоставить все клавиши и кнопки, которые могут запускать это событие.

Чтобы получить доступ к **сопоставлениям ввода**, в меню **Level Editor** выберите **Edit > Project Settings...** и в разделе **Engine** выберите параметр **Input**.

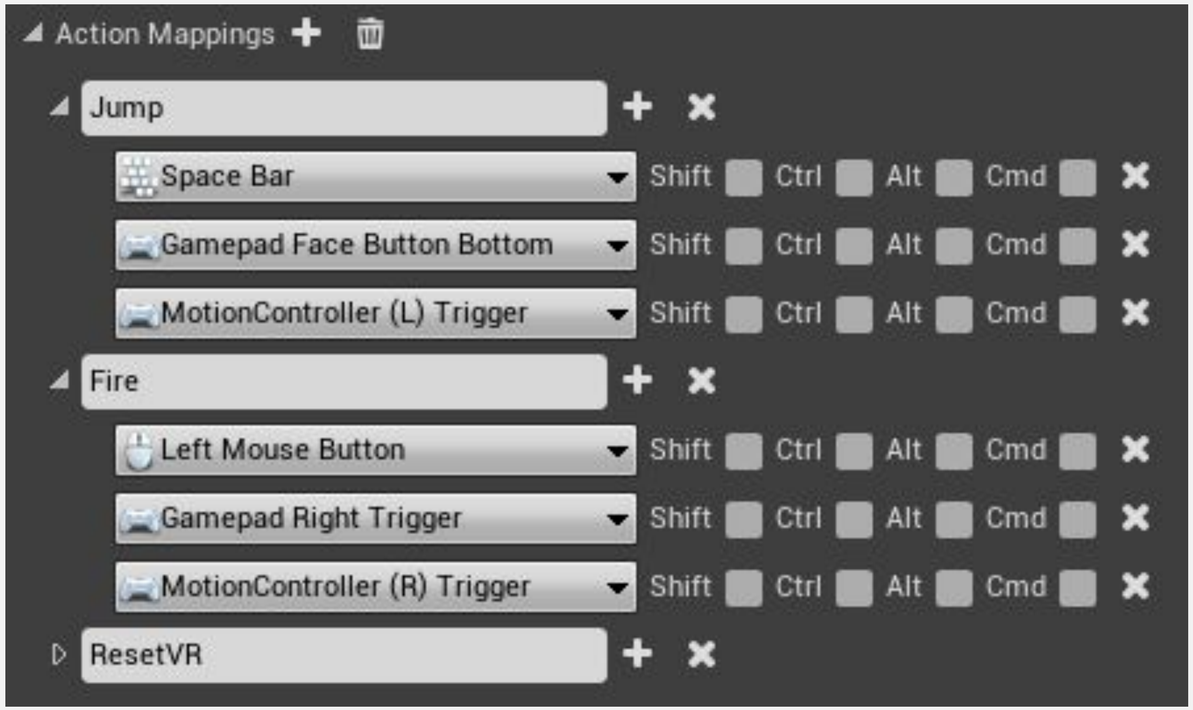


СОПОСТАВЛЕНИЯ ДЕЙСТВИЙ

Сопоставления действий предназначены для нажатия и отпускания клавиш и кнопок.

На изображении справа показан пример сопоставления действий из шаблона от первого лица First Person.

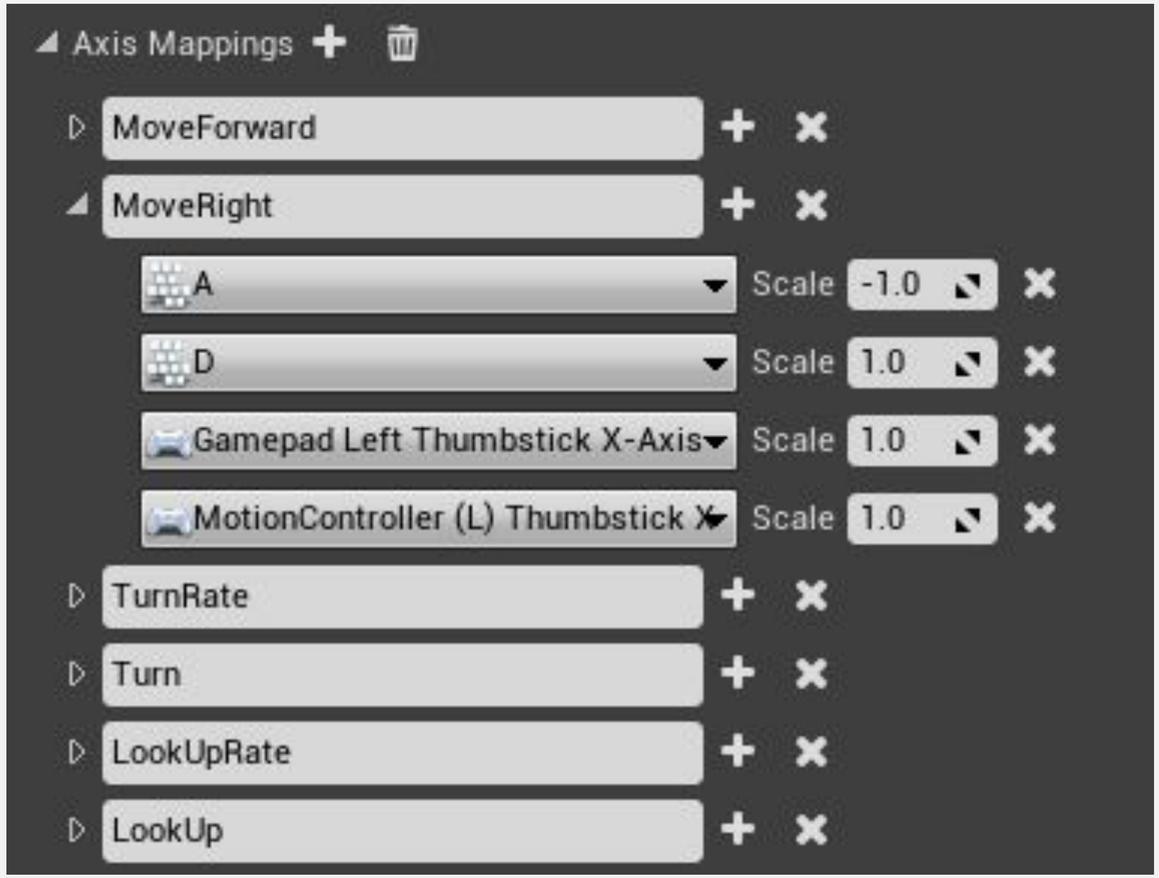
В этом примере было создано действие с именем «**Jump**», которое может запускаться **клавишей пробел**, нижней лицевой кнопкой геймпада или левым триггером контроллера движения.



СОПОСТАВЛЕНИЯ ОСЕЙ

Сопоставления осей **Axis Mappings** допускают вводы, которые имеют непрерывный диапазон, например движение мыши или аналоговых джойстиков геймпада.

Клавиши и кнопки также могут использоваться при отображении оси. В примере справа действие **MoveRight** сопоставляется с клавишей «**D**» со значением свойства **Scale**, установленным на «**1.0**», и с клавишей «**A**» со значением, установленным на «**–1.0**», что представляет собой обратное направление.

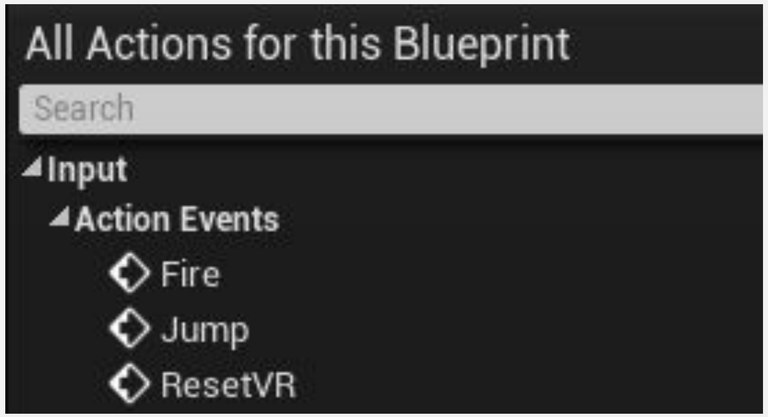


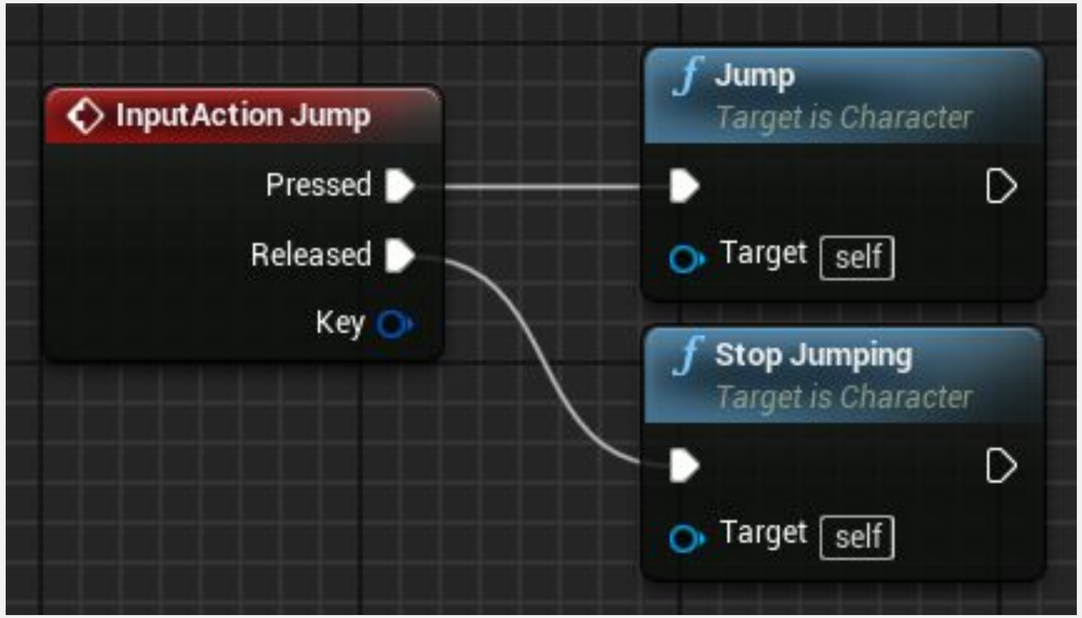
СОБЫТИЯ INPUT ACTION

Все сопоставления действий **Action Mappings** доступны в редакторе **Blueprint Editor** в разделе **Input> Action Events** в контекстном меню **context menu**.

Событие **InputAction** создается при нажатии или отпускании связанных с ним клавиш или кнопок.

На нижнем изображении справа показан пример события **InputAction**.



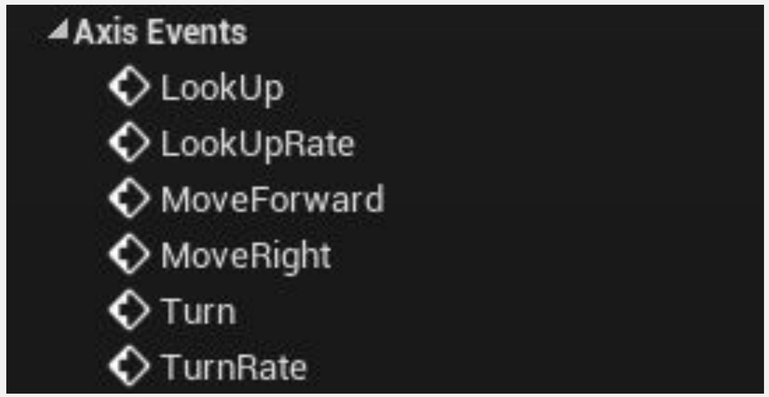


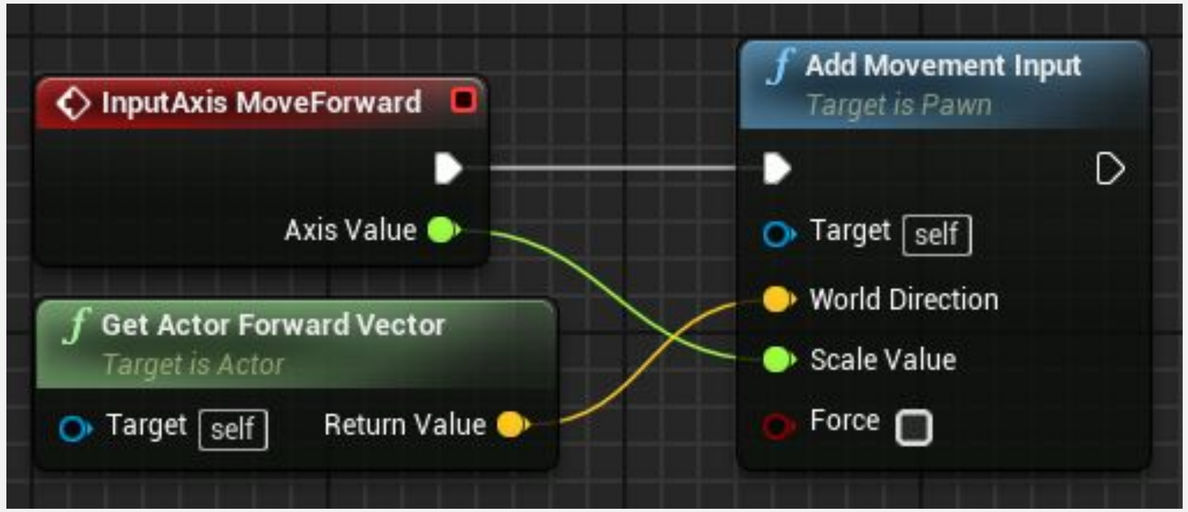
СОБЫТИЯ INPUT AXIS

Все сопоставления осей **Axis Mappings** доступны в редакторе **Blueprint Editor** в разделе **Input> Axis Events** в контекстном меню **context menu**.

Событие **InputAxis** постоянно сообщает текущее значение оси.

На нижнем изображении справа показан пример события InputAxis.





ИТОГИ

В этой лекции были представлены компоненты и Construction Script, а также показано, как добавлять различные типы событий в Blueprint.

Также было рассмотрено, как управлять экземплярами Actor и как выполнять сопоставления ввода.