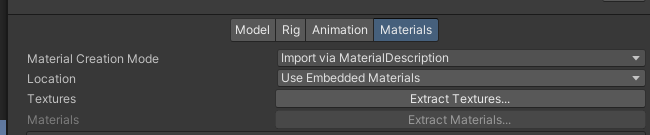
Первое что нужно сделать это импортировать модель в юнити, обычным переносом, но после этого будут скорее всего проблемы с текстурами, необходимо на модели нажать 2 кнопки:



Нужно по очереди нажать Extract Textures и Extract Materials, и именно в этом порядке, иначе потом придется все прокидывать вручную.

Дальше добавим компонент аниматора и разберем его настройки:

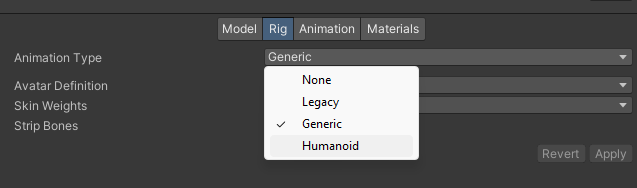
Culling mode – Указывает как будет вести себя аниматор когда он покинет область видимости камеры:  
Always Animate – Не будет выключать анимацию, и все время будет воспроизводить ее, стоит оставлять только на главных объектах, что-бы не терять в производительности.  
Cull update transform – Отключит анимацию, но будет производить расчеты, что-бы в момент появления объекта на камере это выглядело так будто он не переставал двигаться.  
Cull completely – Отключит анимацию и не будет производить никаких расчетов.

Update mode – То, как будет происходить обновление анимации, грубо говоря, в каком методе, там на выбор есть 3 варианта:  
Normal – Анимация будет обновляться чуть позже Update и немного раньше Late update.  
Animate Physics – Будто наша анимация находиться в Fixed update.  
Unscaled time – На нашу анимацию не будет влиять время, то есть Time.scale – никак не будет воздействовать на нашу анимацию.

Apply root motion – Будет ли анимация влиять на передвижение объекта, чаще всего следует отключать эту галочку.

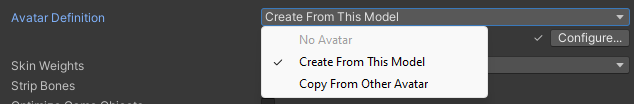
Настрйока персонажа.

Если наш персонаж Humanoid – тоесть похож на человека, его нужно конвертировать в человека.



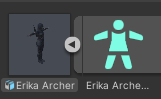
Это в настройках модели.

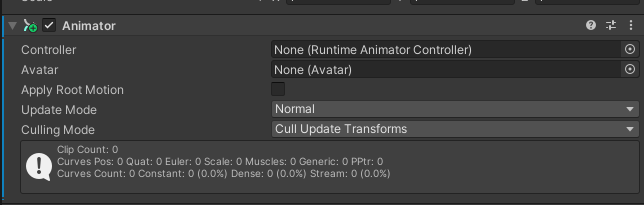
Так-же нужно выбрать, создать аватар или использовать уже имеющийся, важно, если мы используем уже имеющийся, то позиция, наименование и расположение костей должно быть одинаковым.



Первый вариант создаст аватар на основе этой модели, а второй на основе уже готового аватара, естественно там надо будет указать на основе какого аватара мы хотим сделать наш тип.

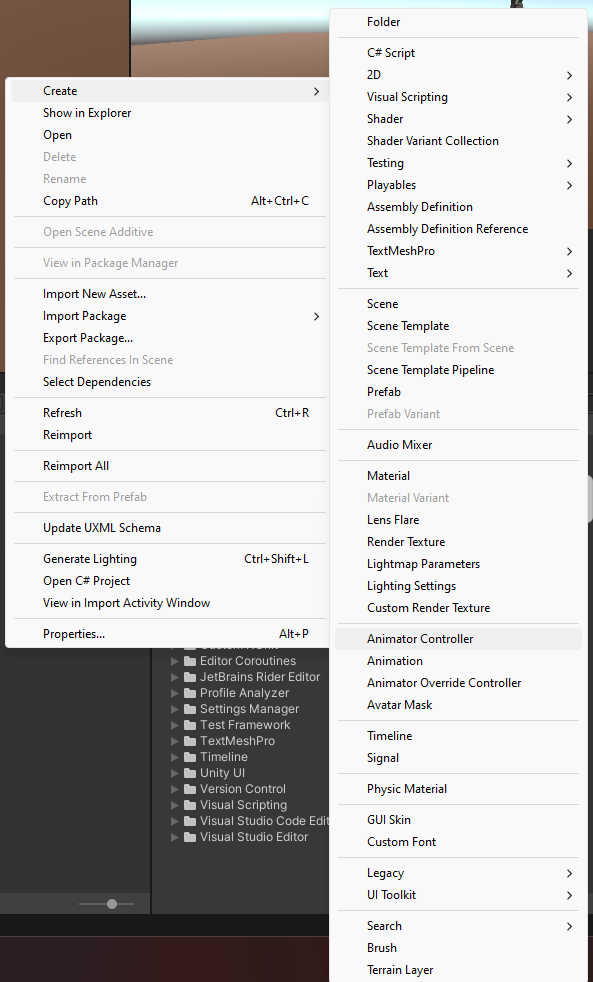
После этих манипуляций в игроке автоматически сгенерируется аватар, его необходимо поместить в анимейшен контроллер, который находиться на объекте, к которому мы хотим применить эту анимацию.





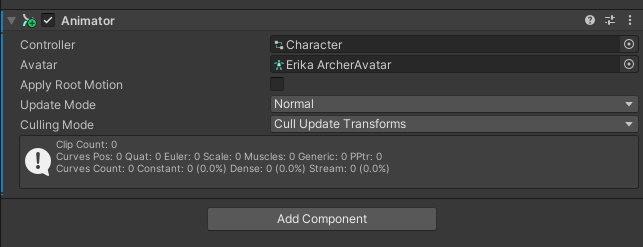


И в первый пункт необходимо добавить анимейшен контроллер, который уже необходимо создать вручную, через окно проджект.



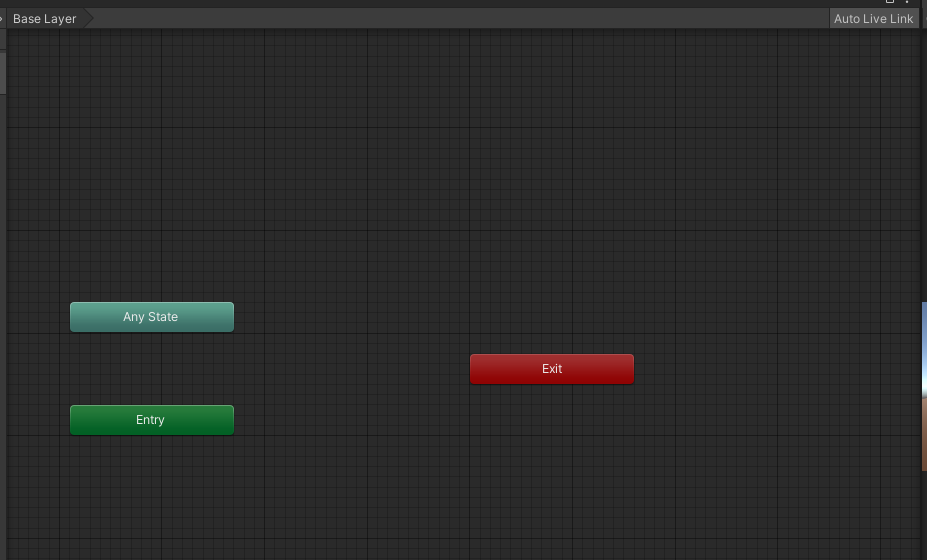
И добавить его в компонент аниматора.

Теперь наш компонент аниматора полностью настроен!



Аниматор.

2 раза кликаем на компонент контроллера, который мы создали и поместили в аниматор, и попадаем в окно настройки анимаций.



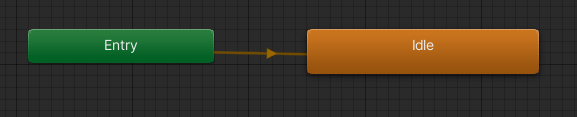
Entry – Точка входа в анимации, от нее будет все стартовать.

Any state – Переход в анимацию из любой возможной анимации.

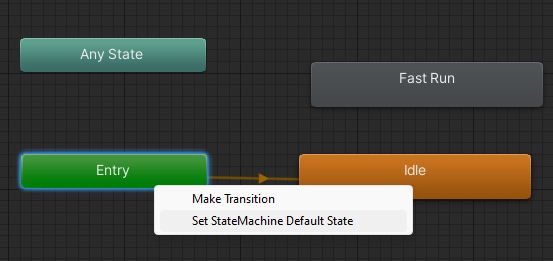
Exit – Выход из анимации после чего опять запуститься Entry.

Для добавления анимаций мы можем либо перетащить уже имеющиеся, либо создать тут пустышку и добавить в нее уже имеющиеся анимации.

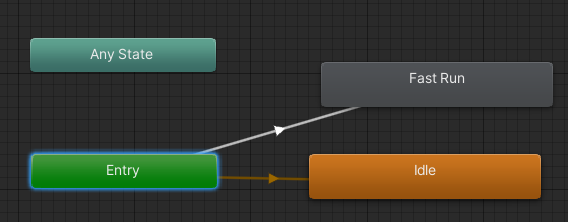
Я добавил 2 анимации, к первой у меня автоматически подвязался Entry.



Это будет всегда, точка входа всегда должна иметь анимацию в которую перетекает.

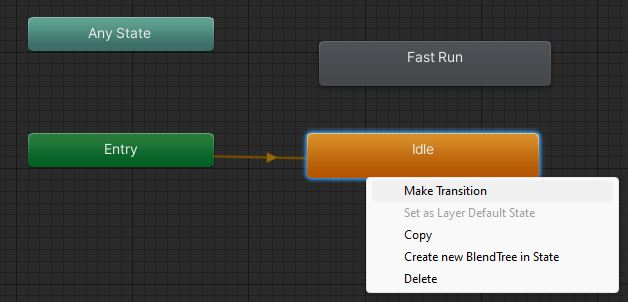


Ее можно переключить, если нажать ПКМ по энтри и выбрать второй пункт, который устанавливает дефолтную анимацию.



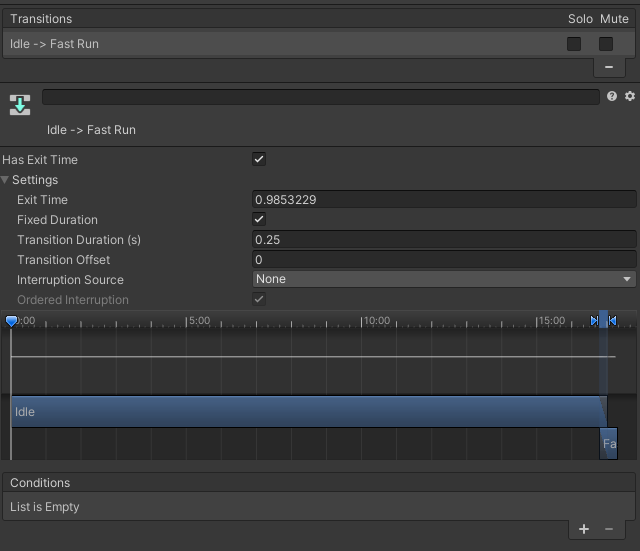
После чего появляется стрелочка, которой мы должны указать новую анимацию для энтри.

Таким-же способом мы можем объеденять анимации между собой, к примеру если ПКМ по Идле, и выбрать Make transition – То появится стрелочка которой можно указать, в какую анимацию она будет перетекать.

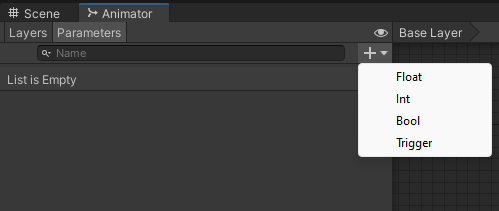




Если нажать на эти стрелочки, то можно увидеть несколько настроек:



Has exit time – Означает что переход сработает когда анимация закончиться  
Дальше идут настройки перехода одной анимации в другую, грубо говоря, чем больше одна анимация заходит в другую тем плавнее будет выглядеть переход.  
И в конце, Conditions – Условия перехода, которые можно добавить в окне параметров.

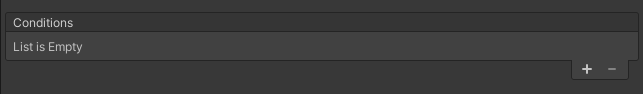


Тут есть несколько вариантов перехода. Числовые будут случаться при достижении определенного числа, бул – когда мы выставим флаг, а тригер, когда мы его щелкнем.

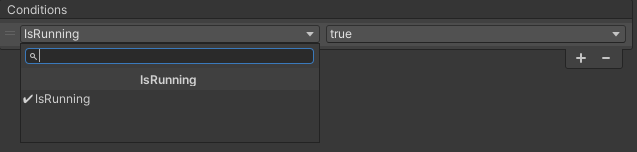
К примеру, я создал булевый переход



Первым делом его надо добавить к стрелочкам, в условия перехода.



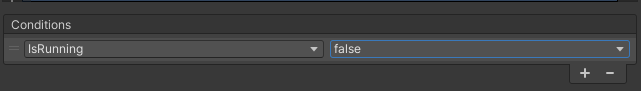
Нажимаем на +, и добавляем тот что нам нужен.



Поскольку выбора у нас нет, то остаемся на одном, дальше говорим, какое значение должен иметь наш флаг для перехода, оставим Тру.



А в стрелочке обратной выставим фалсе.

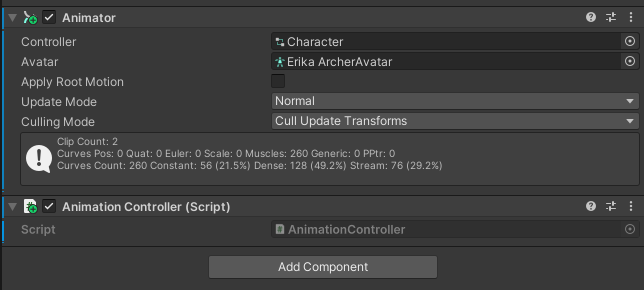


И не забудем убрать с обоих стрелочек галочку Has exit time, чтобы наша анимация не переходила по завершению.

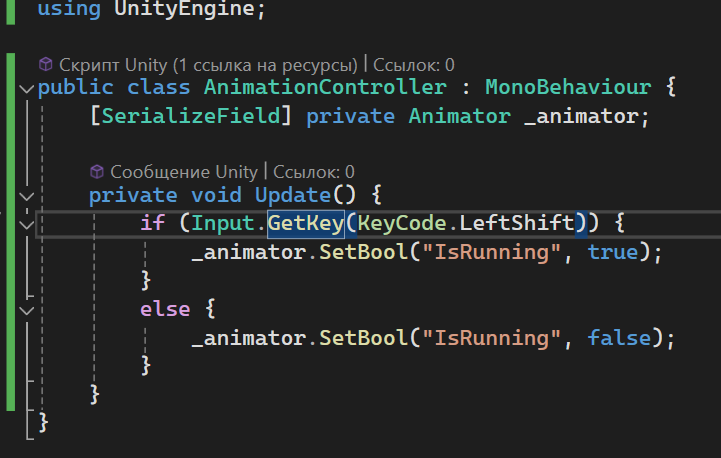


Теперь при нажатии на галочку у нас человек побежит, а при выключении будет останавливаться.

Чтобы нажимать эту галочку из кода, нам понадобиться ссылка на аниматор и скрипт.



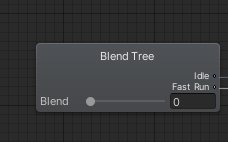
Скрипт есть. Осталось его написать.



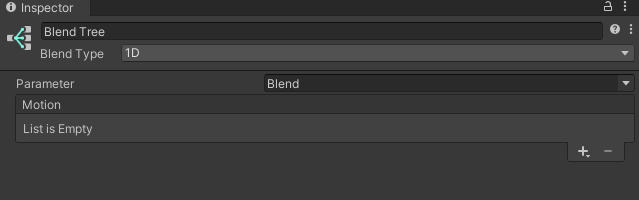
Вот таким простым скриптом можно управлять анимациями, в идеале создавать энум под каждый параметр анимации, так как при использовании энума будет достаточно поменять его в одном месте, это на тот случай, если мы допустим ошибку в стринге.

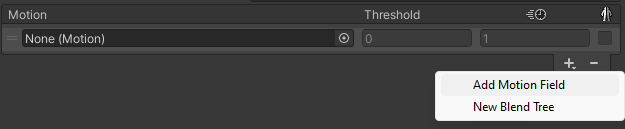
Параметры Set, есть не только для бул, но и для всех возможных переходах, таких как Int, float, trigger.

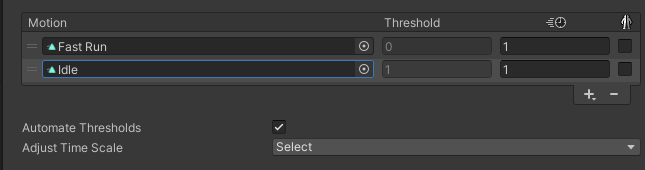
Для более плавной анимации рекомендуется использовать Blend tree. Создать его так-же как Emty state, но выбрав уже Blend tree, дальше 2 раза нажать на него и попасть в настройки.



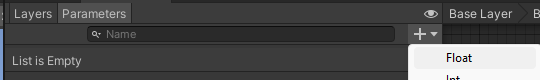
Нажав на нее, справа появиться окно настроек куда и можно добавить наши анимации через +







Я добавил наши 2 анимации, для их настройки уже бул не подойдет, нужно создать числовую переменную, флоат.



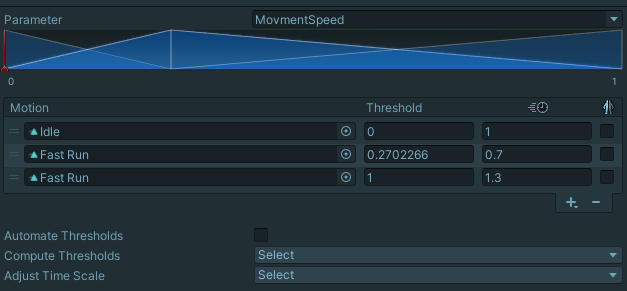


И выбрать ее в параметрах blend tree.



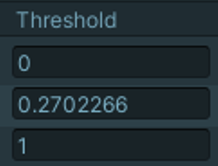
Теперь изменяя ее из кода, мы будем изменять состояние анимации.

Если оставить просто 2 анимации, то это очень будет похоже на стандартное смешивание, если мы хотим сделать что-то сложнее, то нужно добавить еще одну или 2 анимации, к примеру бега и поиграться с параметрами скорости воспроизведения и смещения относительно друг друга.



К примеру, так будет более плавный разгон.

Начало относительно входящего значения.



Скорость воспроизведения анимации.

