Unreal Engine 4 Tutorial
Blueprint scripting – The runner

Preparation:

For this tutorial the following components are necessary:

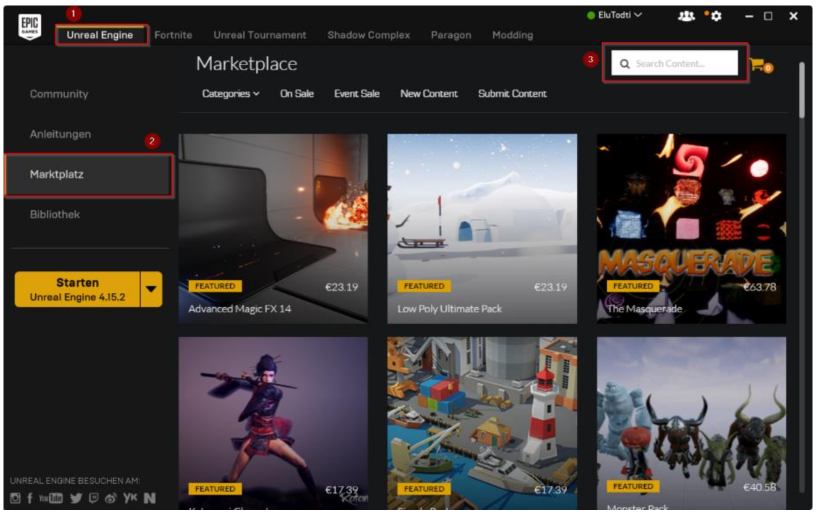
- Unreal Engine 4:

  Get the Unreal Engine 4 for free here: www.unrealengine.com
  Just create an account, download the Epic Games Launcher and there you can get the Unreal Engine 4.

(The following components you will only need if you want to do the additional part)

- Animation Starter Pack:

  The Animation Starter pack is available for free on the marketplace.
  In the Epic Games Launcher click on
  1 „Unreal Engine“
  2 „Marketplace“
  3 Search



- Mixamo Character Pack:

  ~~The Mixamo Character Pack is available for free on the marketplace, too.~~

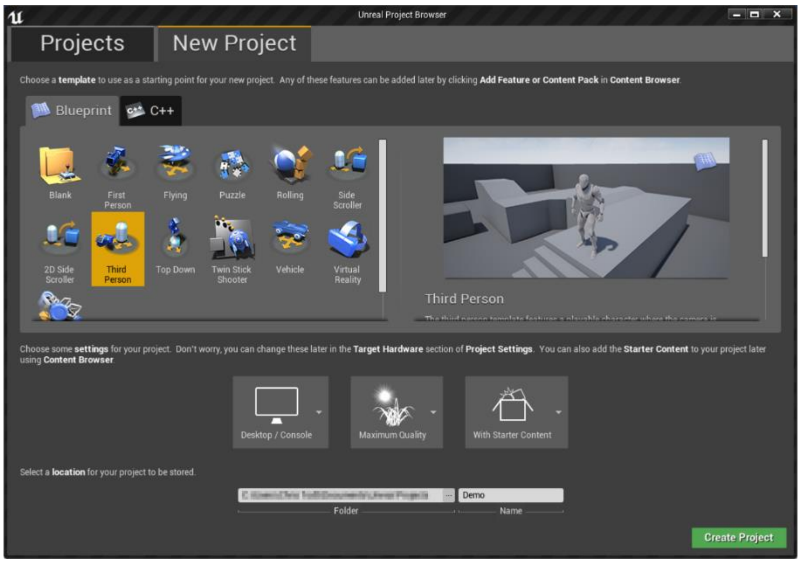
  Meanwhile the Mixamo Character Pack isn't compatible with the Unreal Engine 4 version anymore. – instead download the character I created for this tutorial here: https://github.com/EluTodti/Unreal-Engine-4-Presentation/tree/master/Body. It's created with MakeHuman and exportet to a .fbx file.

Note: This tutorial was created for Unreal Engine 4 4.15.3

Start the Unreal Engine.

Create a new Project with the Blueprint „Third Person“. Choose the settings in the picture beside.

Some short camera controls are listed unten.



Choose a **template** to use as a starting point for your new project. Any of these features can be added later by clicking **Add Feature or Content Pack** in **Content Browser**.

Choose some **settings** for your project. Don't worry, you can change these later in the **Target Hardware** section of **Project Settings**. You can also add the **Starter Content** to your project later using **Content Browser**.

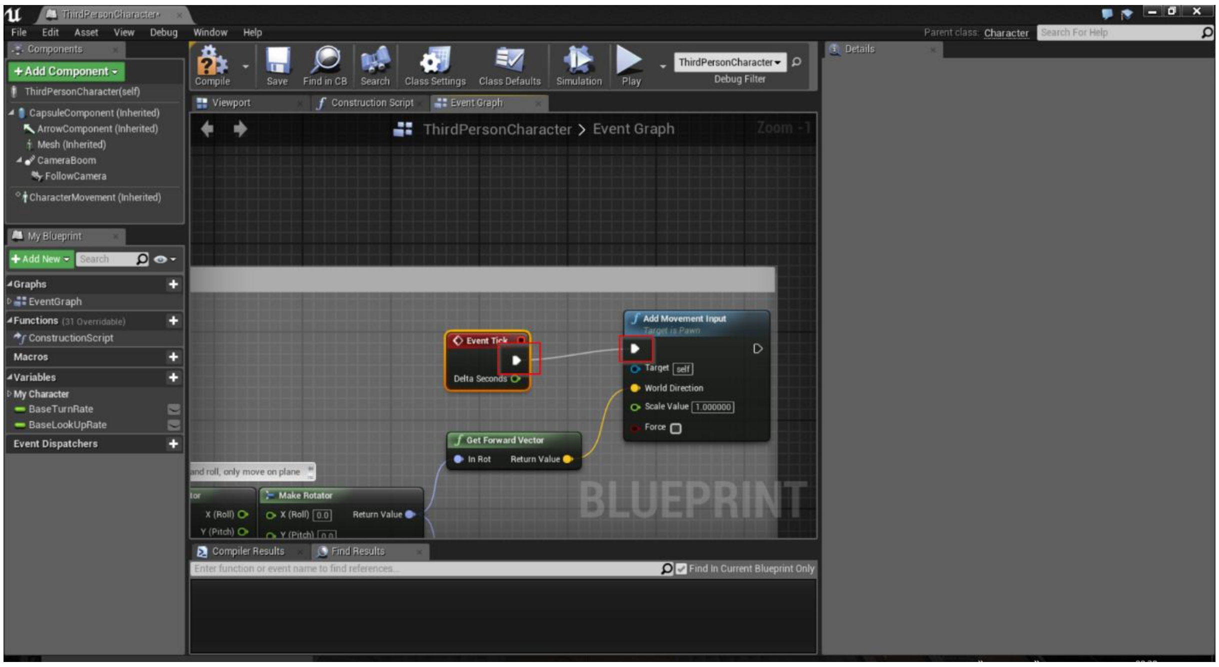
Select a **location** for your project to be stored.

In the ThirdPersonBP folder head to Blueprints and double click the Third Person Character.

In Components select Camera Boom. Then in Details „Transform“, on the right, set the z location higher to have a better overview (look in the Viewport tab).

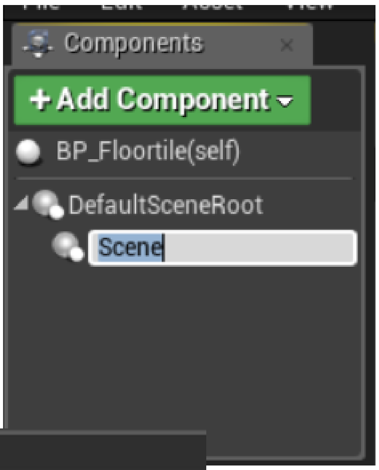
To make the character move constantly forward, head to the „Event Graph“ and delete „InputAxis Move Forward“ and replace it with an „Event tick“ (right click) – connect the exec path.

Delete anything but Jump and Movement input.

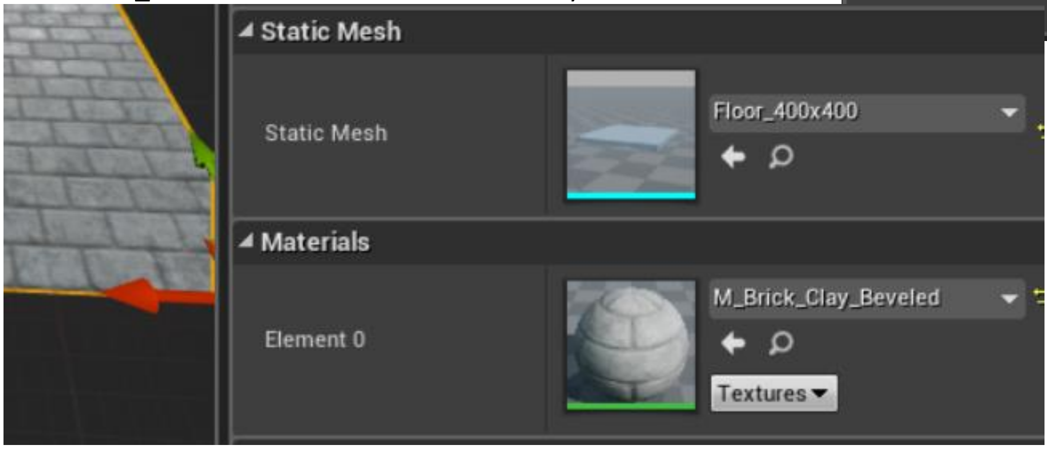


Building the track

To build a track that keeps spawning endlessly, we'll need a new Blueprint Class. To create one, right-click in the Content Browser and click Blueprint Class. Make it an Actor and name it „BP_Floortile". (you have to return to the tab „ThirdPersonExampleMap")

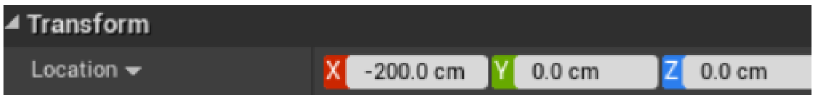


Head to its editor (double-click) and, under Components, add the component „Scene". Make the Scene the new DefaultSceneRoot by dragging it to the top. Add the component „Cube". As Static Mesh, selectFloor_400x400 and as material whatever you wish.


Static Mesh
Static Mesh: Floor_400x400
Materials
Element 0: M_Brick_Clay_Beveled

You should place the axes as shown in the picture above.

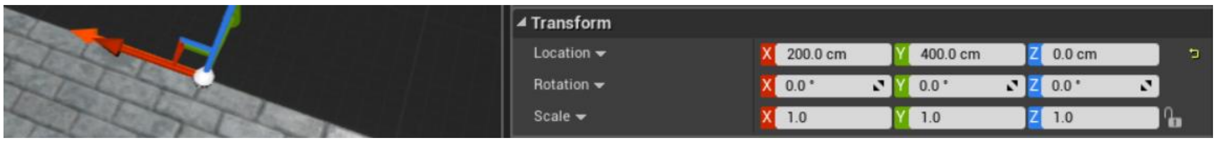
Set the location as the following:


Transform
Location X -200.0 cm Y 0.0 cm Z 0.0 cm

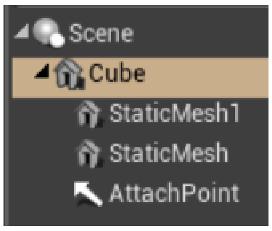
Now, duplicate the Floor and scale it to receive 2 walls, one on each side.
(select and press Strg + c and then Strg + v) (scale – see Transform on the right)

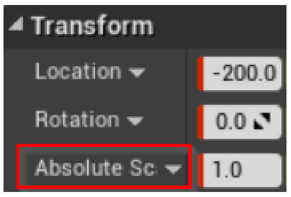
To be able to spawn one Floortile after another, we'll need an Attach Point. For this we can simply use an arrow. Add an arrow component and move it to the front-middle of the Floortile. It'll need some detailed config in the Transorm field. Name the arrow „AttachPoint".

Ignore the direction of the arrow. The only thing that matters is the start of the arrow.


Transform
Location X 200.0 cm Y 400.0 cm Z 0.0 cm
Rotation X 0.0 ° Y 0.0 ° Z 0.0 °
Scale X 1.0 Y 1.0 Z 1.0

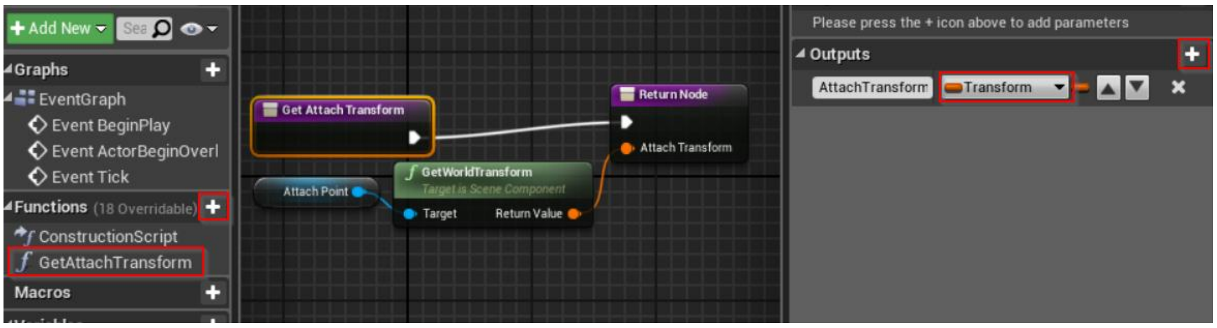
Please note: It's very important to your components ordered like in the picture on the right and to change the Scaletype of the Floor relative to the World(Red rectangle on the right).


Scene
Cube
StaticMesh1
StaticMesh
AttachPoint


Transform
Location -200.0
Rotation 0.0
Absolute Sc 1.0

Next, we need a new function and name it „GetAttachTransform". Make it have an Output with a Transform variable, which is to be named AttachTransform. Starting from a „get AttachPoint" connect this to a „GetWorldTransform" method and return the methods value to the return node of Get Attach Transform.

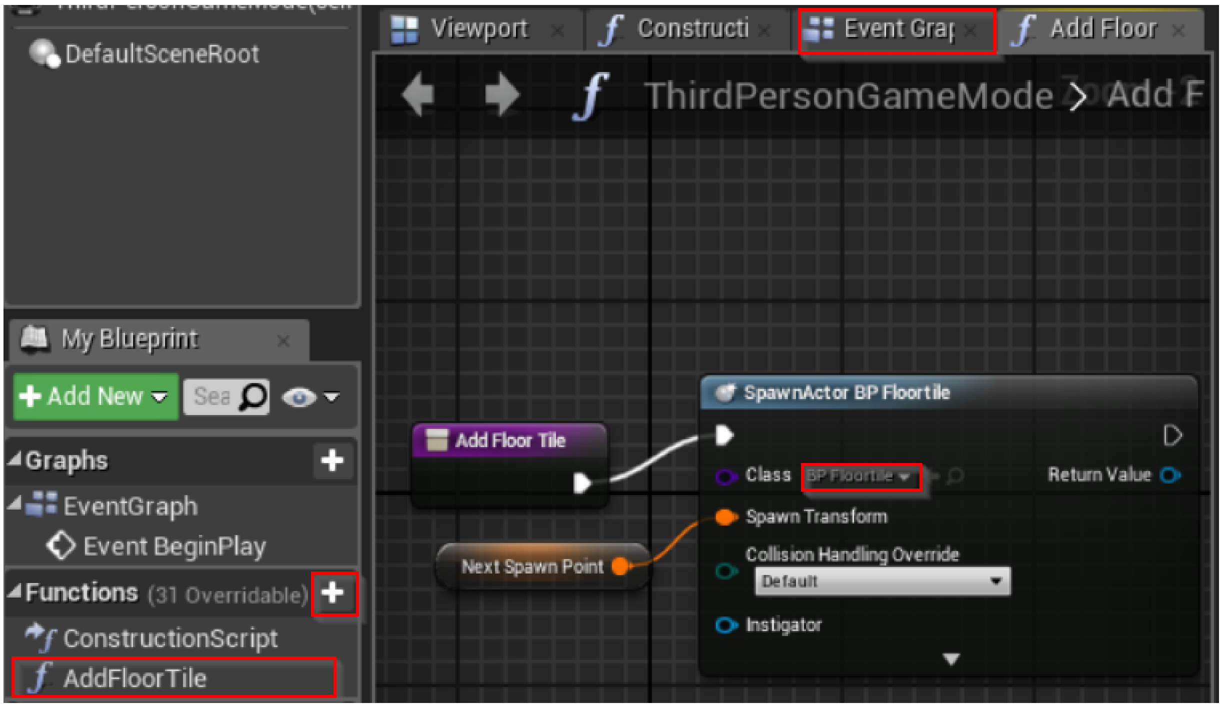
This is used to find out where the arrow is and where a new Floortile has to be attached.



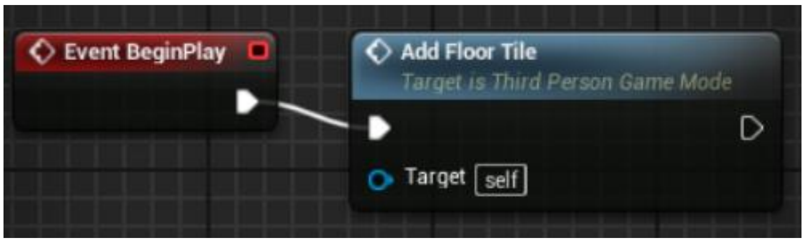
Compile the BP_Floortile and head beack to the Unreal Editor.

Create a new level (CTRL-N or File...) and use the Default. Simply reuse the existing game mode by double-clicking it and access the blueprint via „Open Full Blueprint Editor".

Add a new function and name it „AddFloorTile". Connect it to a „SpawnActorFromClass" method and change the class to „BP_Floortile". At SpawnTransform, add a new variable and name it „NextSpawnPoint".



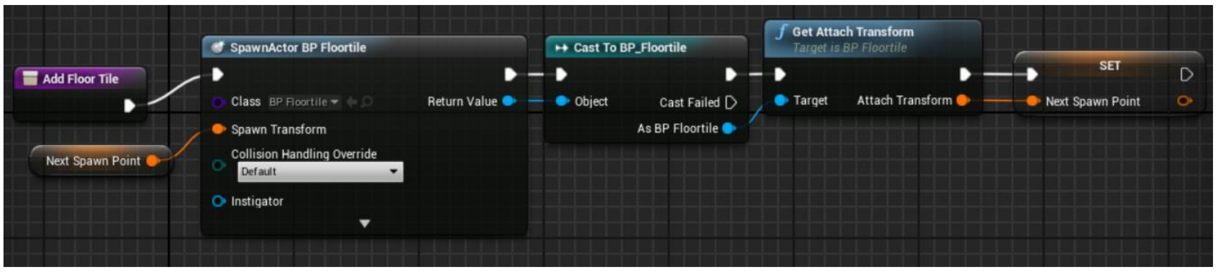
Go to the EventGraph tab and add a „EventBeginPlay“ event. Connect this event to the function „AddFloorTile“.



Back in the Unreal Editor, delete the default floor. Press Alt+S to simulate what we just typed in.

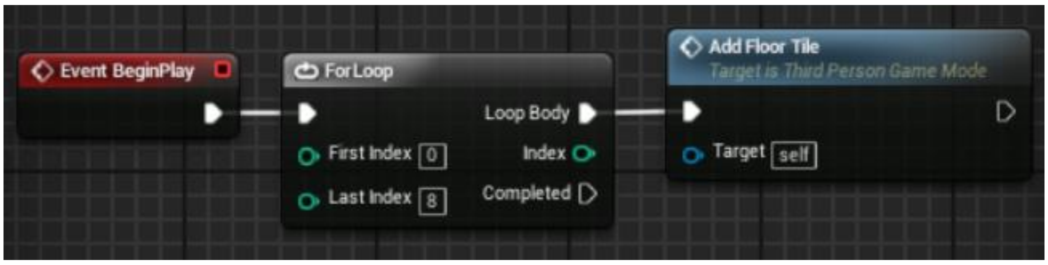
Stop the simulation and move the „Player Start“ a little bit forward. (note: you may have to rotate it to get the right direction).

Head back to the game mode (double-click in Unreal Editor) to the AddFloorTile tab. Cast the return value of the SpawnActorBP_Floortile to BP_Floortile with the „CastToBp_Floortile“ function. From this function call the „GetAttachTransform“ function, which needs to know where to attach the next tile. So we'll tell it by setting the „NextSpawnPoint“. When the function is called again it uses this NextSpawnPoint.

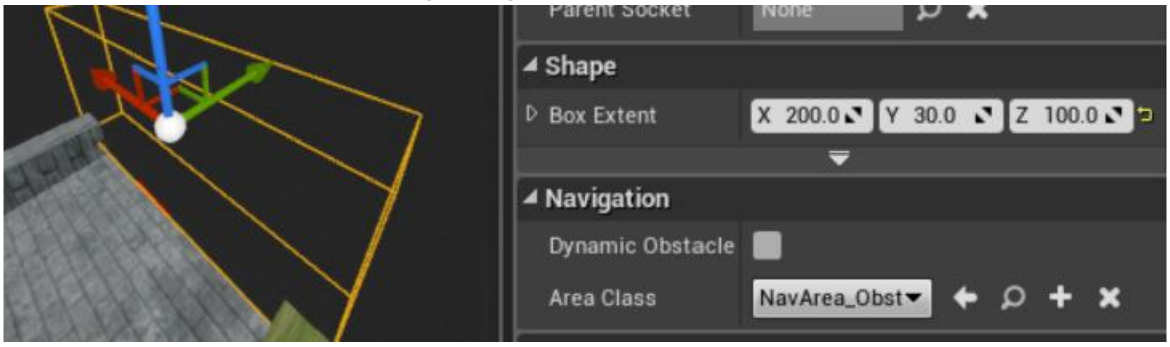


Go to the EventGraph tab.

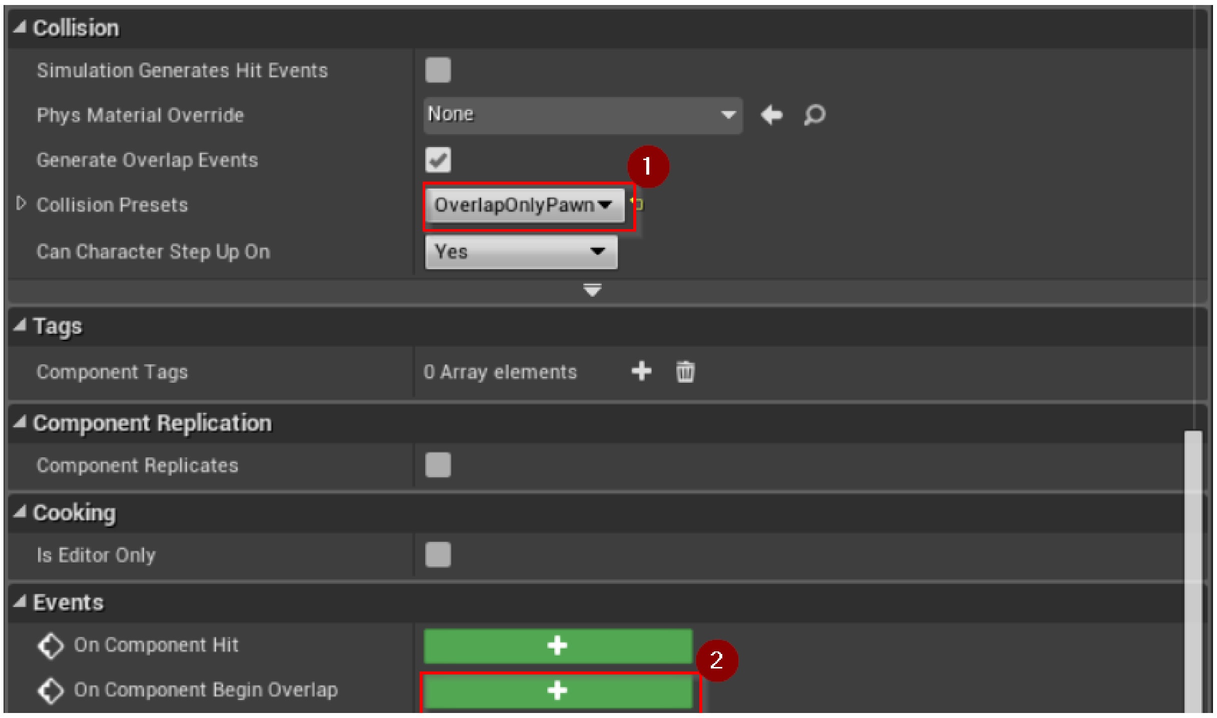
We'll insert a loop here to create multiple Floortiles behind each other. After this, there will be 9 FloorTiles. Don't forget to compile.



But for our game, we'll need more than just 9 tiles. But with an endless loop it will end in a crash of the game. So we will delete the tiles we passed and then spawn new ones. Switch back to the BP_FloorTile. There, add a BoxCollision and name it EndTrigger. Move it to the end of the floortile (where the arrow is) and make it big enough to cover the whole tile.

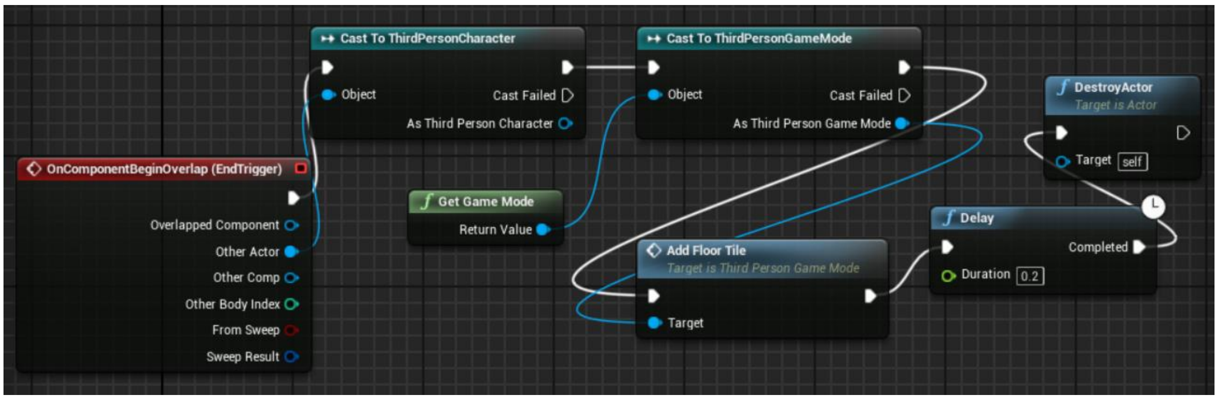

Parent Socket None
Shape
Box Extent X 200.0 Y 30.0 Z 100.0
Navigation
Dynamic Obstacle
Area Class NavArea_Obst

Set the Collision Presets of the EndTrigger to „OverlapOnlyPawn“ and create a „On Component Begin Overap“ event by clicking the plus (see below).


Collision
Simulation Generates Hit Events ☐
Phys Material Override None
Generate Overlap Events ✓
Collision Presets OverlapOnlyPawn
Can Character Step Up On Yes
Tags
Component Tags 0 Array elements
Component Replication
Component Replicates ☐
Cooking
Is Editor Only ☐
Events
On Component Hit
On Component Begin Overlap

Fort he Overlap event, we need to check if it’s our character by using the „Cast To ThirdPersonCharacter“. If it is, get the game mode („GetGameMode“function) cast it to the ThirdPersonGameMode(our game mode – „CastToThirdPersonGameMode“) and add a FloorTile. To do so, connect the CastToThirdPersonGameMode function to an „AddFloorTile“ function.

Through this, there still will be a endless number of tiles. So we have to delete the old tiles. Insert a „Delay“ (so our character can’t fall down). After the delay, insert a „DestroyActor“ function to delete the tile.



!!! In the OnComponentBeingOverlapped function one must use the „OtherActor“ to connect to the cast.!!!

To test it out, drag a ThirdPersonCharacter on the first tile.

Further movement

For testing first. Do this at the default map of the ThirdPersonCharacterBP.

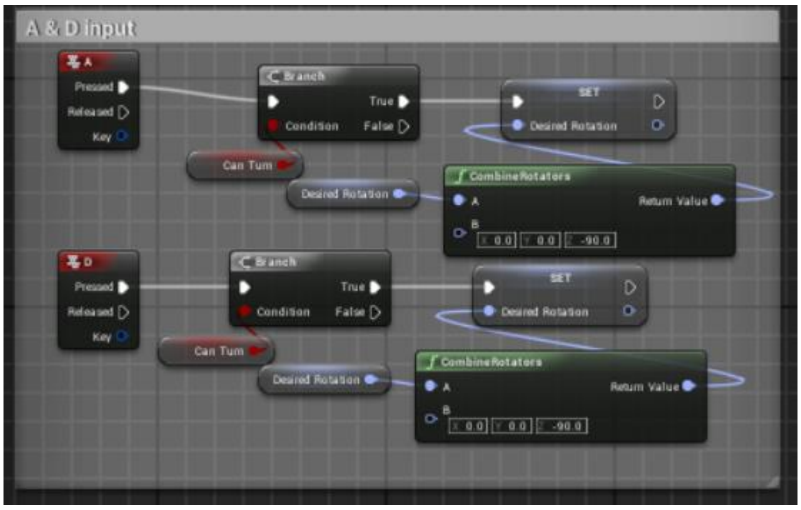
Now we want to make the character turn.

Add the Keyboard Events „A“ and „D“ and a new variable bool (on the left – „Variables +“). Name the bool „CanTurn“, drag it tot he A Event and select „get“.

Add a „Branch“ (right-click) to connect A and CanTurn. Create an additional variable, name it „DesiredRotation“ and change the type to „rotator“.

We will need a setter of this variable and a getter connected with „CombineRotators“. This gets the actual rotation and sets it according to the CombineRotators method. Return the value of this method tot he setter.

Don’t forget to connect the „Branch“ with the setter.

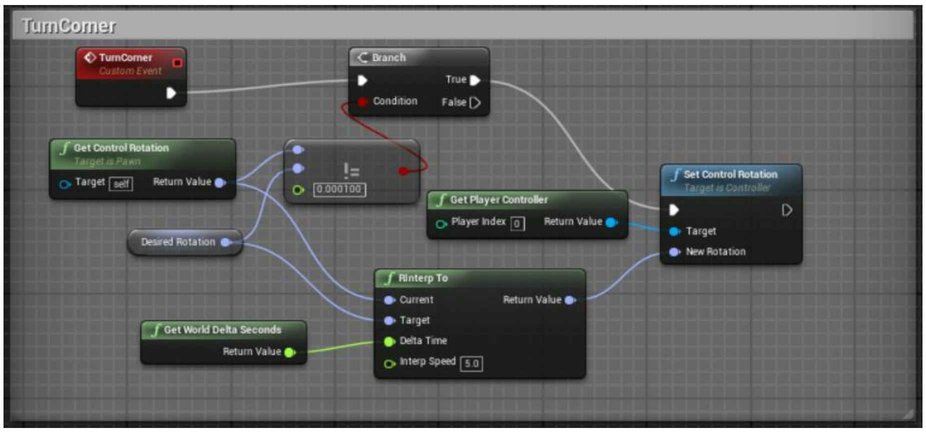


To make the character react to these inputs properly, we’ll have to insert a new method between our existing Event Tick and the Add Movement Input.

Create a custom event (right-click) „TurnCorner“. Now we have to get the actual rotation and compare it with the desired rotation. If they are not equal the character should turn.

To achieve this we need the „Get Control Rotation“ method and our „DesiredRotation“ Variable compare on „Not Equal“ and branch those with the „TurnCorner“ event.

Then, to set the rotation, we need to a „Get Player Controller“ method to „Set Controller Rotation“. The new rotation should be our DesiredRotation. To do so, we’ll have to interpret the

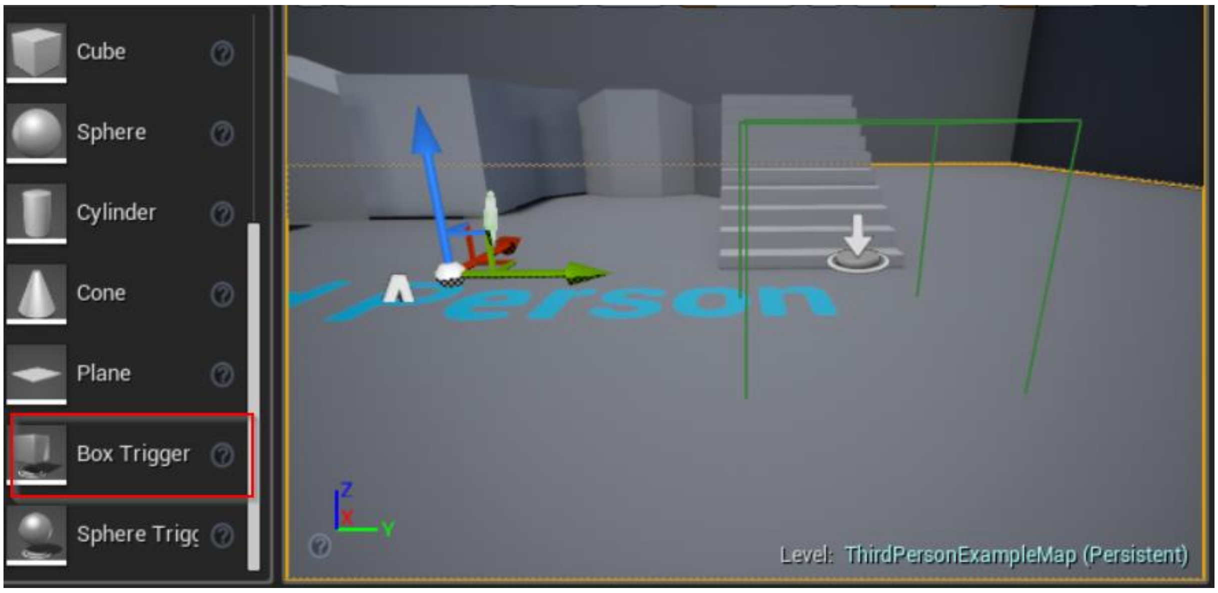


GetControllerRotation tot he DesiredRotation using the „RInterpt To“ method. Add the „GetWorldDeltaSeconds“ and set the Interp Speed a little higher to make it look smoother.

Now insert the new TurnCorner between the EventTick and the AddMovementInput. To test it, set the default value of CanTurn to true (select the value and check the Default Value box – don't forget to set it back to default → false)

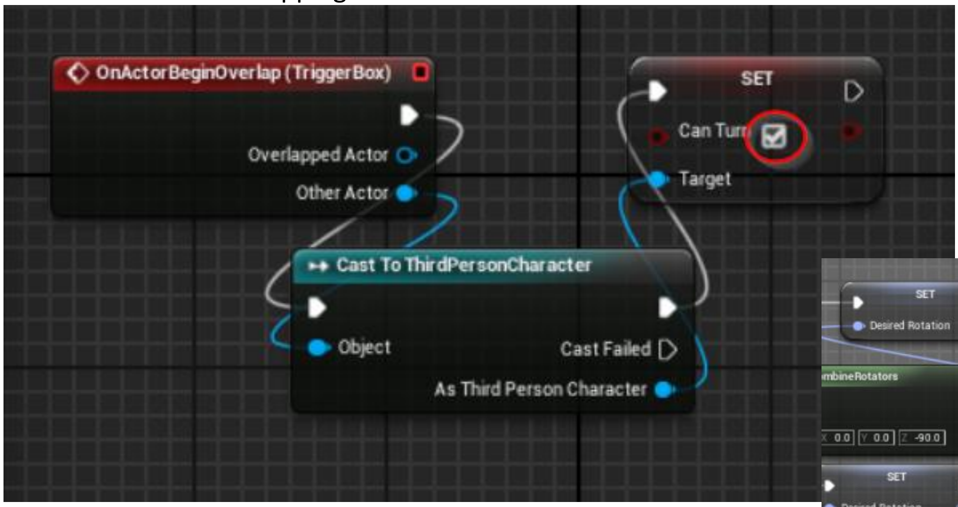
As we don't want the character to turn at all time we need some trigger.

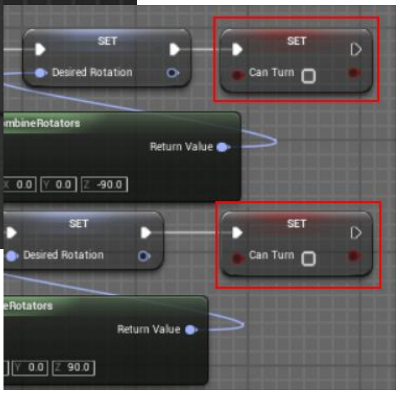
Go back to the Unreal Editor and drag a „BoxTrigger“ in front of the character and scale it a little bit bigger (transform on the right). If you want to see it ingame, set the „Actor Hidden In Game“ to false.



Now, to le tour character do something when he collides with this BoxTrigger, open the Level Blueprint. You can find it on the middle top on dropdown of the Blueprints.

In the Unreal Editor, select the BoxTrigger. Now switch tot he Level Blueprint, right-click, and create an „Add On Actor Begin Overlap“ event. Connect it with „Cast To ThirdPersonCharacter“ and then connect „As ThirdPersonCharacter“ with a setter of „CanTurn“. Check the box next to Can Turn to set te bool to true if overlapping.


OnActorBeginOverlap (TriggerBox)
Overlapped Actor
Other Actor
Cast To ThirdPersonCharacter
Object
Cast Failed
As Third Person Character
SET
Can Turn
Target



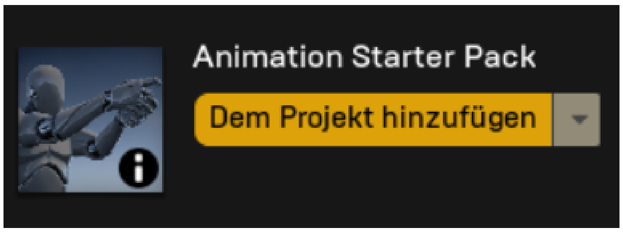
To only once accept the A or D press, add a setter of CanTurn after the A and D inputs (open the Third Person Character in the Content Browser)

Additional part:
Animation Retargeting

I will try to complete this part as soon as I got the time as it seems tob e a little bit more complicated ☺

In the Launcher under library add the Animation Starter Pack to your project.

In the content browse, create a new folder, right click and click „Import to“ and select F.fbx (the file you downloaded).



Camera control

Short overview of camera controls:

- Right-click: Turn camera horizontally & move for/backward
- Left-click: Move camera in every direction
- Right- & Leftclick: Move camera horizonzally and vertically

Note:

I made my tutorial according to this video tutorial https://docs.unrealengine.com/latest/INT/Videos/PLZlv_N0_O1gbY4FN8pZuEPVC9PzQThNn1/yS-yQfo0lc0/index.html