Creating Waypoints in UE4

Chaima Jemmali

Version 2.1

March 2016

Overview:

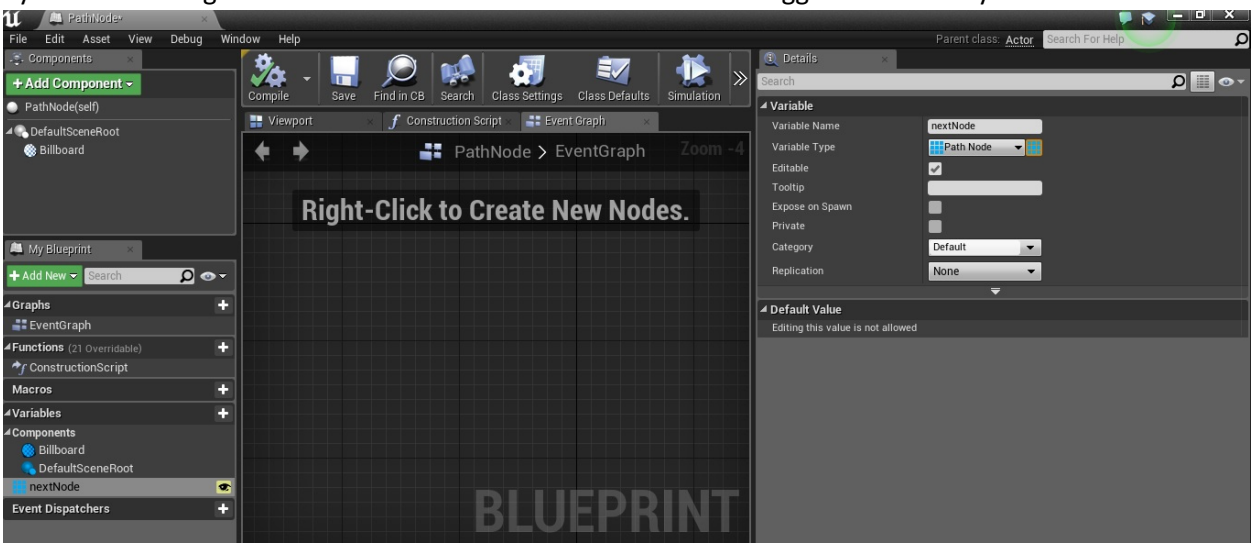
This tutorial is designed to walk you through two methods of creating Nodes (waypoints) in UE4. The waypoints can be used to code a pathfinding algorithm, such as A*.

Details:

Method 1 – Blueprint start

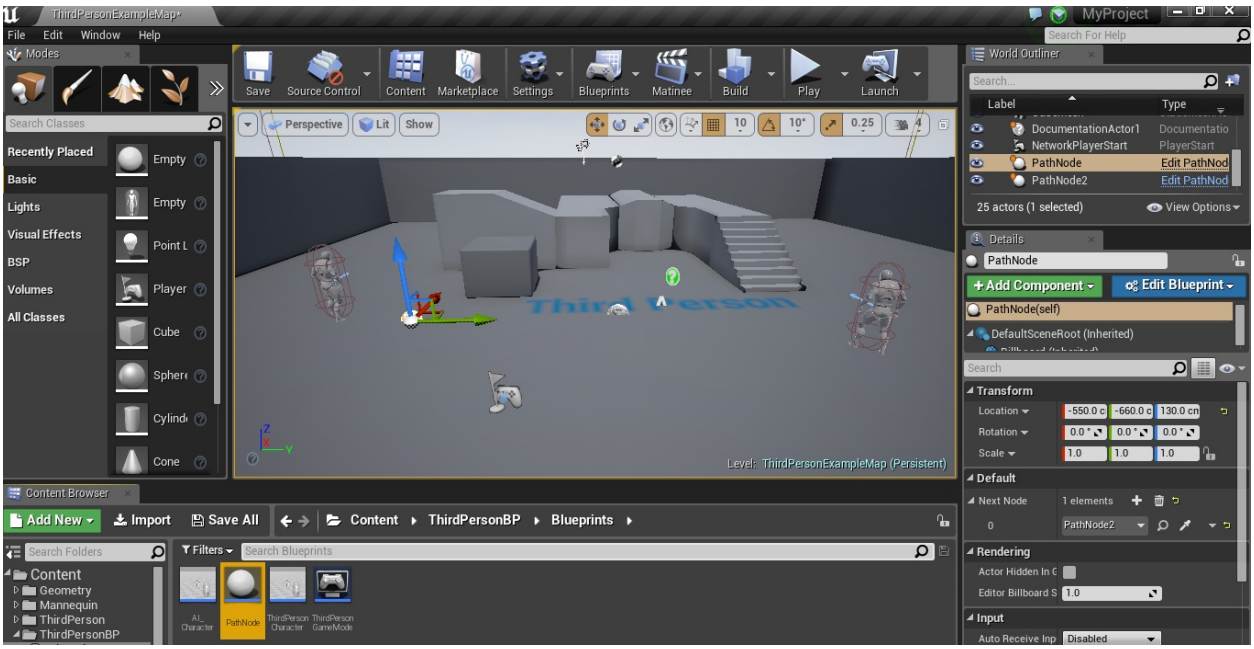
This first method creates a waypoint starting from a Blueprint. Such Blueprints are easy to make and convenient.

1. Create a Blueprint class (Actor). Name it PathNode.
2. Open the PathNode Blueprint and create a new Variable called nextNode. Change the type of the variable for a reference of PathNode and make the variable public (visible) by clicking the eye next to it. Right click on the icon to the left of the name to toggle it to an array.



Compile and save.

3. Go to the editor and place two PathNodes. Under the details panel under Default, make the nextNode variable of each Node reference to the other Node.



4. Create an Actor C++ class named MapActor. In this class we are going to iterate over all the actors in our world that contain “PathNode” in their name (those correspond to our waypoints) and we are going to display their location to the screen.

   In MapActor.cpp, add the following inclusions:

```
#include "EngineUtils.h"
#include <string>
```

   Then Change your BeginPlay() method to look like this

```
void AMapActor::BeginPlay() {
  Super::BeginPlay();

  // Iterate over all Actors in the world.
  for (TActorIterator<AActor> ActorItr(GetWorld()); ActorItr; ++ActorItr)
    AActor *Mesh = *ActorItr;
    // If Actor is a “PathNode”, display location on screen.
    if (ActorItr->GetName().Contains("PathNode"))
      if (GEngine)
         GEngine->AddOnScreenDebugMessage(-1, 5.f, FColor::Yellow,
                 TEXT(""+ActorItr->GetName()+" location " + ActorItr->GetActorLocation().ToString()));
  }
}
```

   The TActorIterator iterates over all the actors in our world. Then we are displaying the location only the ones that contain “PathNode”. You can save the locations of all your nodes in a List and use it as your A* map nodes.

5. Back in the level editor, place an instance of the MapActor class in the level. It can be anyplace.

6. Compile and Play/Launch. You should get output on the screen (yellow text in the upper right) indicating locations of the waypoints when you press play.

Method 2 – C++ start

Waypoints made with Method 1 have the disadvantage of not being able to access the Blueprint variables (e.g., nextNode) from the C++ code. Method 2 starts with C++ and creates a waypoint that does not have this restriction.

1. Create a C++ class that extends Actor and call it NodeActor. In NodeActor.h, to the NodeActor class definition, add the following property at the bottom.

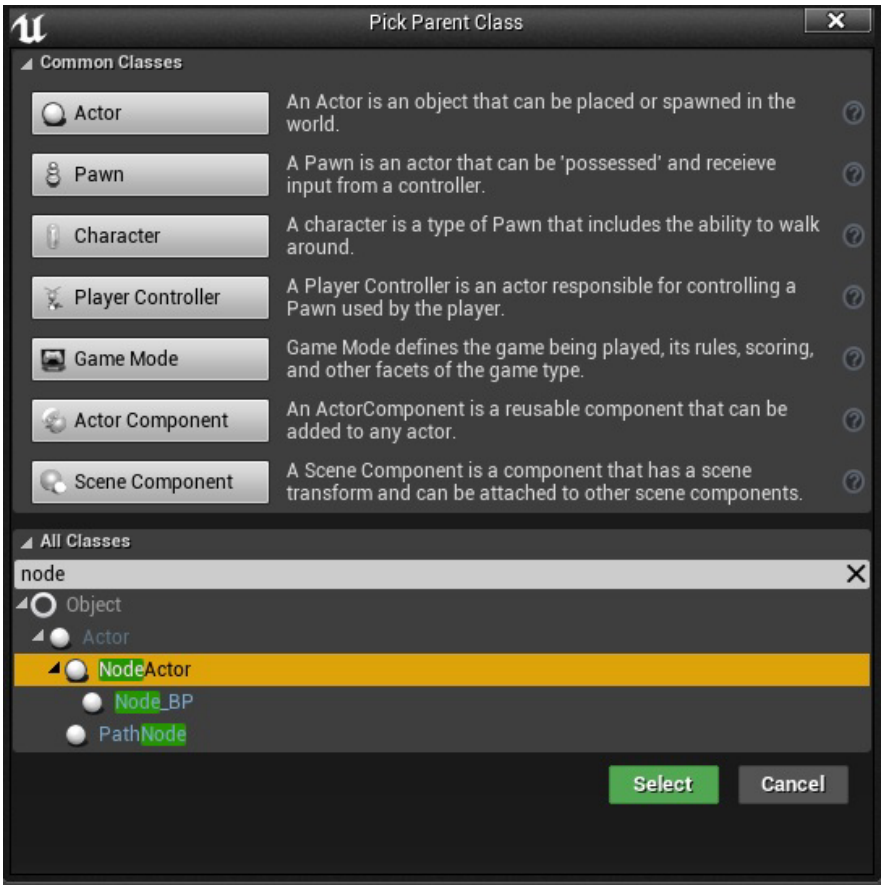
UPROPERTY(EditAnywhere, BlueprintReadWrite)

ANodeActor* nextNode;

The UPROPERTY() macro allows linking of C++ code and features in the Unreal Editor, among other things. UPROPERTY() options can be found here: https://docs.unrealengine.com/latest/INT/Programming/UnrealArchitecture/Reference/Properties/index.html

Compile your code.

2. Create a Blueprint class called Node_BP. The class should extend NodeActor.



Now you can open the editor of Node_BP, you can change the “nextNode” variable at the top.