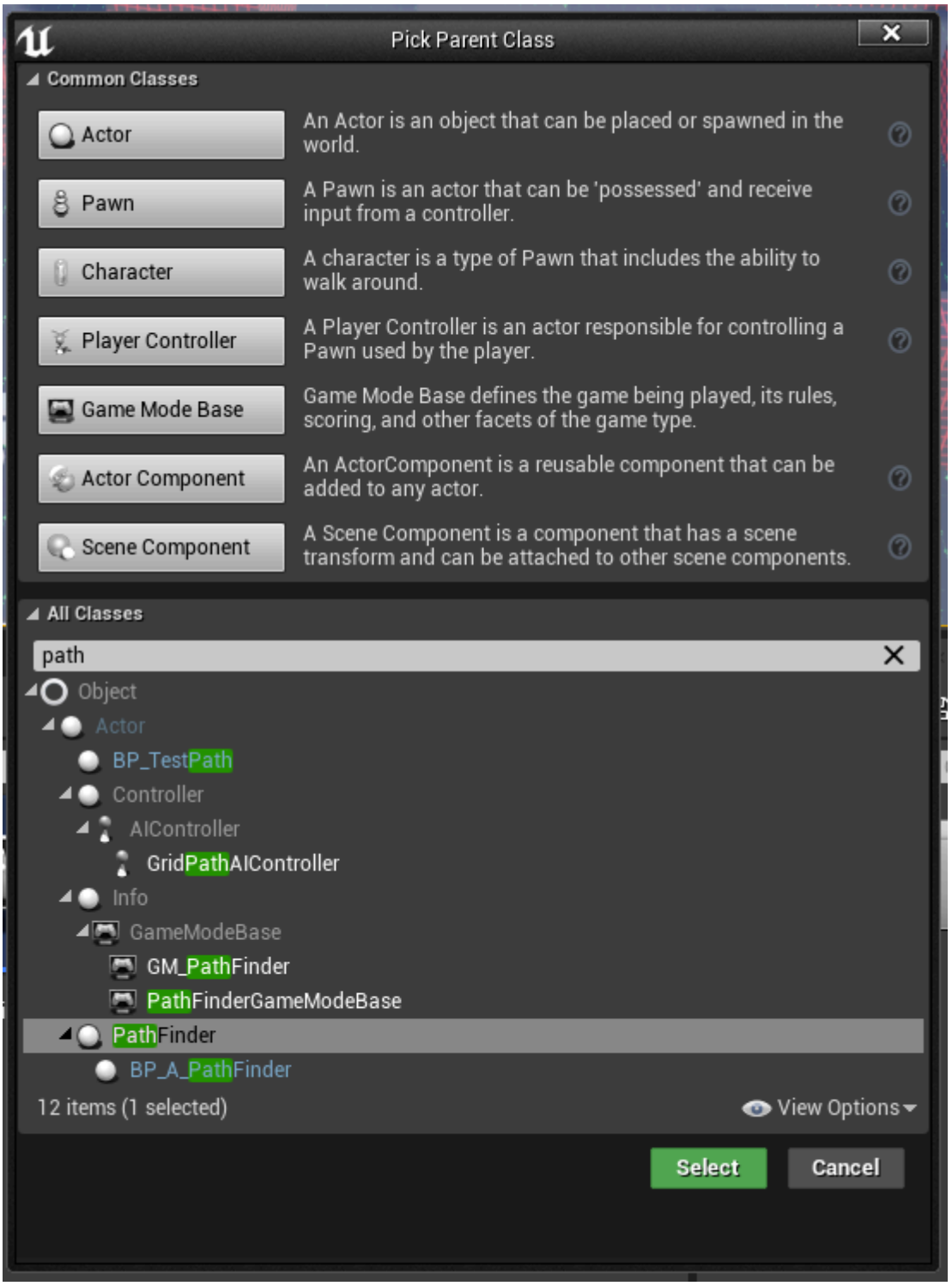
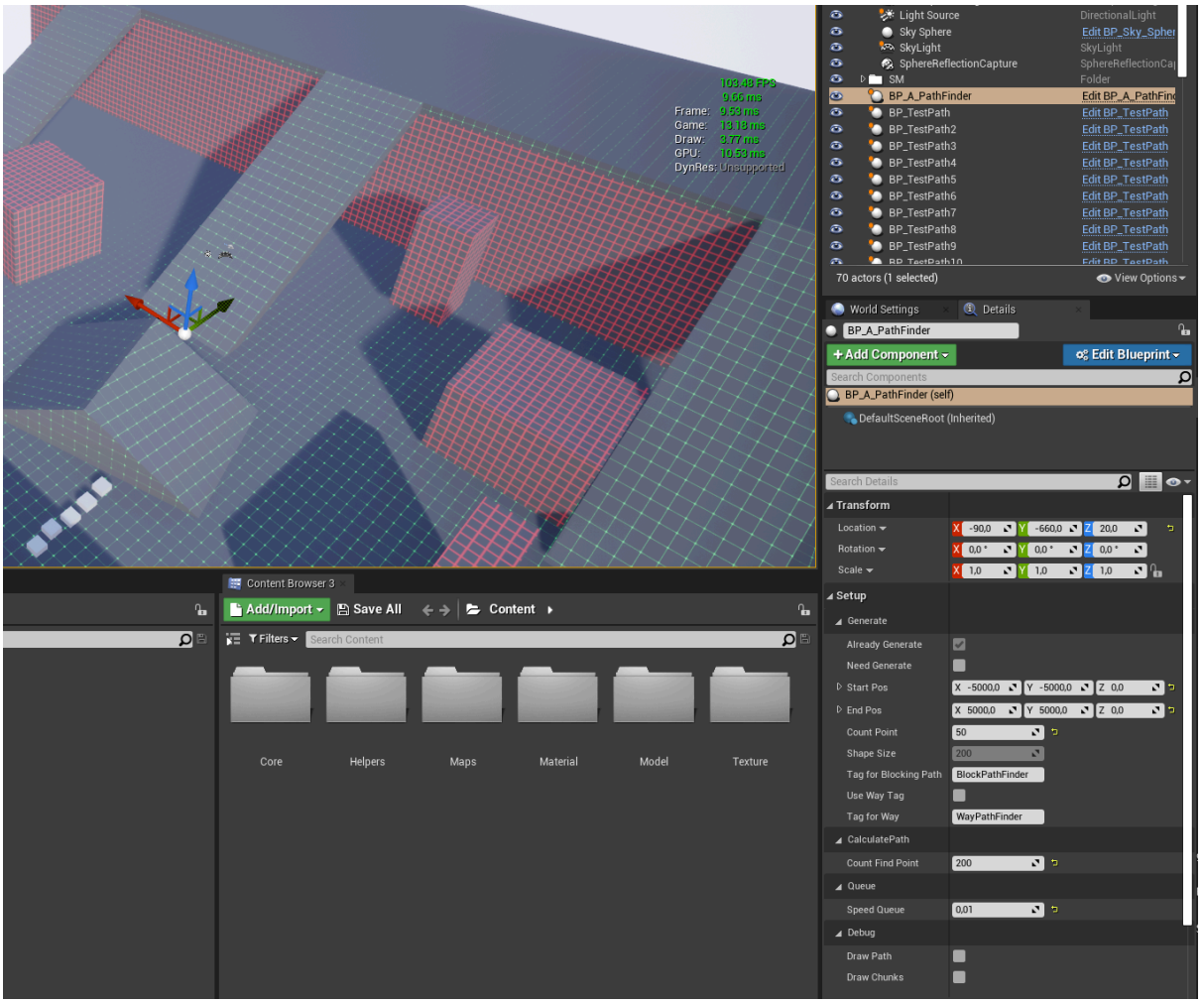
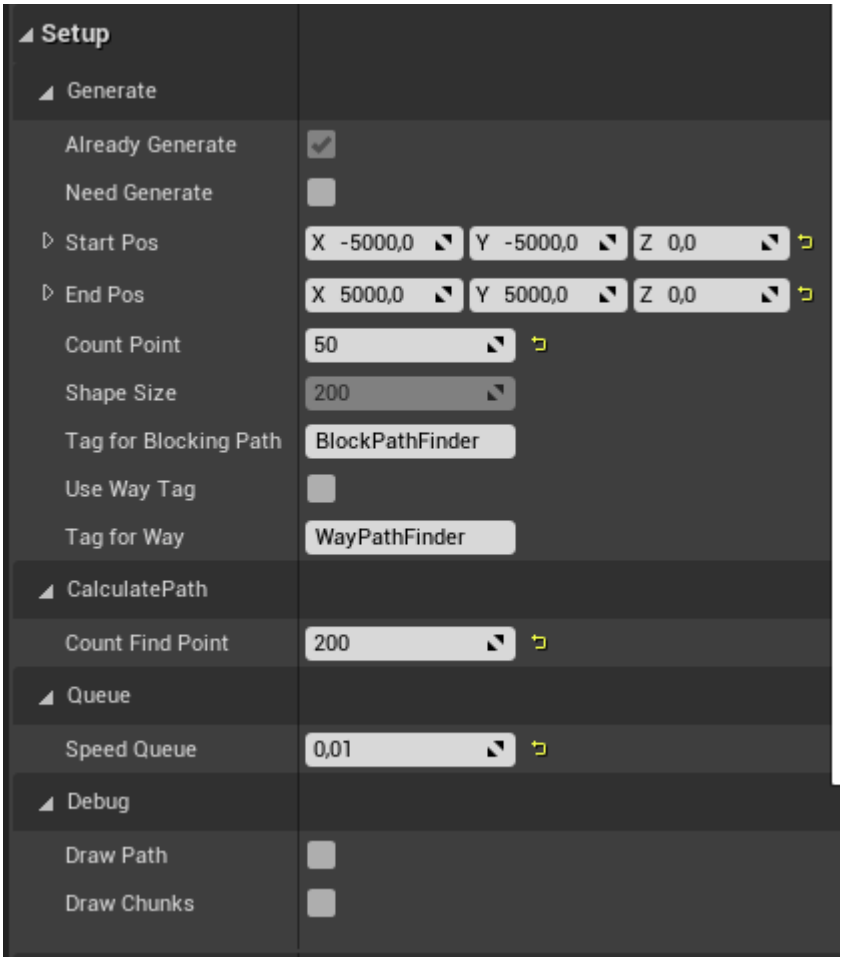
1. Create BP with parent PathFinder and name it like BP_A_PathFinder



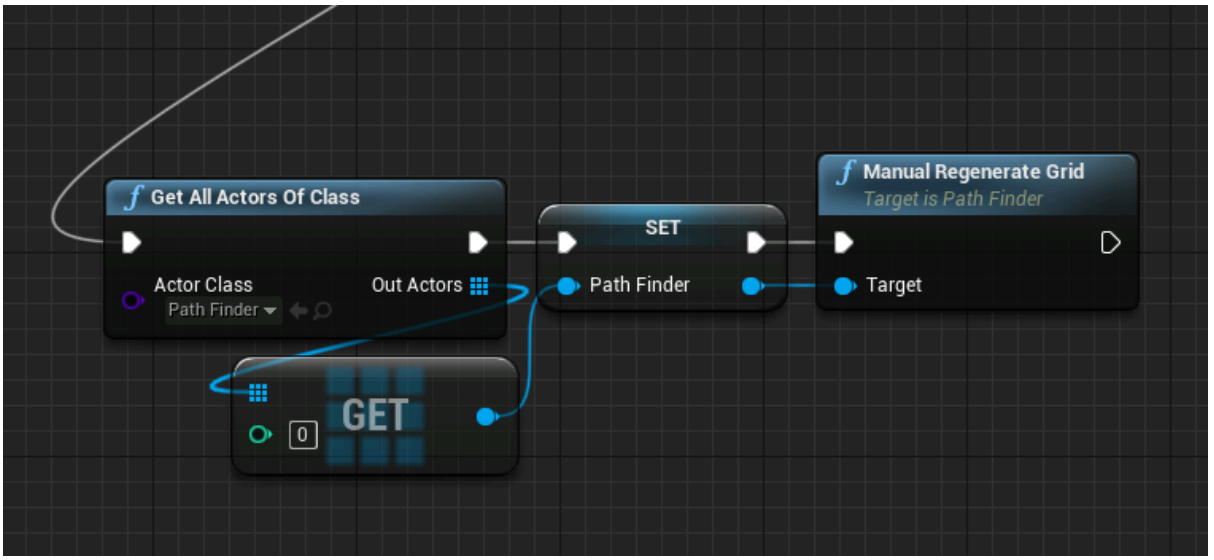
2. Move BP_A_PathFinder to your scene.



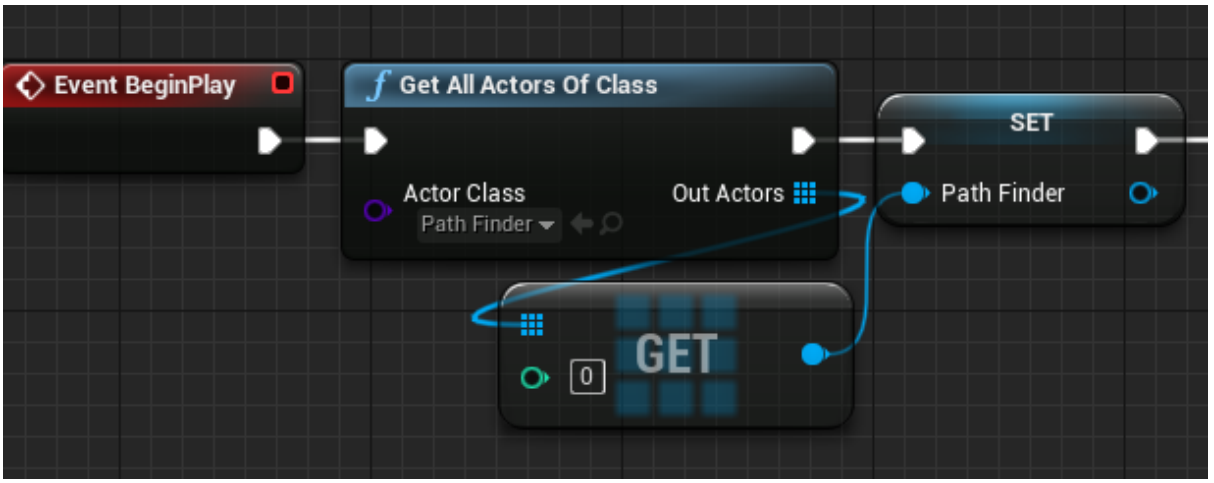
3. Settings:
    a. Need generate - check this check box for generate grid with your settings
    b. Start Pos - World position of start point to generated grid
    c. End Pos - world position of end point to generated grid
    d. Count Point - count by X and Y axes boxes
    e. Tag for Blocking Path - objects with this tag will be considered impassable and the path will be built bypassing these objects. All other objects will be considered suitable for building a path
    f. Use Way Tag - if it is necessary to use the checkbox "Tag for the path", then let it be generated only for objects that have this tag, everything else will be considered impassable.
    g. Tag for Way - tag for the previous item
    h. Count Find Point - the maximum number of iterations that is available for generating a path. If you have reached the limit, the received path will be returned, which was not completed to the destination point.
    i. Speed Queue - speed in seconds. the smaller the number, the faster the queue will pass
    j. Draw Path - after building the path, it will draw it
    k. Draw Chunks - after building the grid, it will draw the resulting chunks

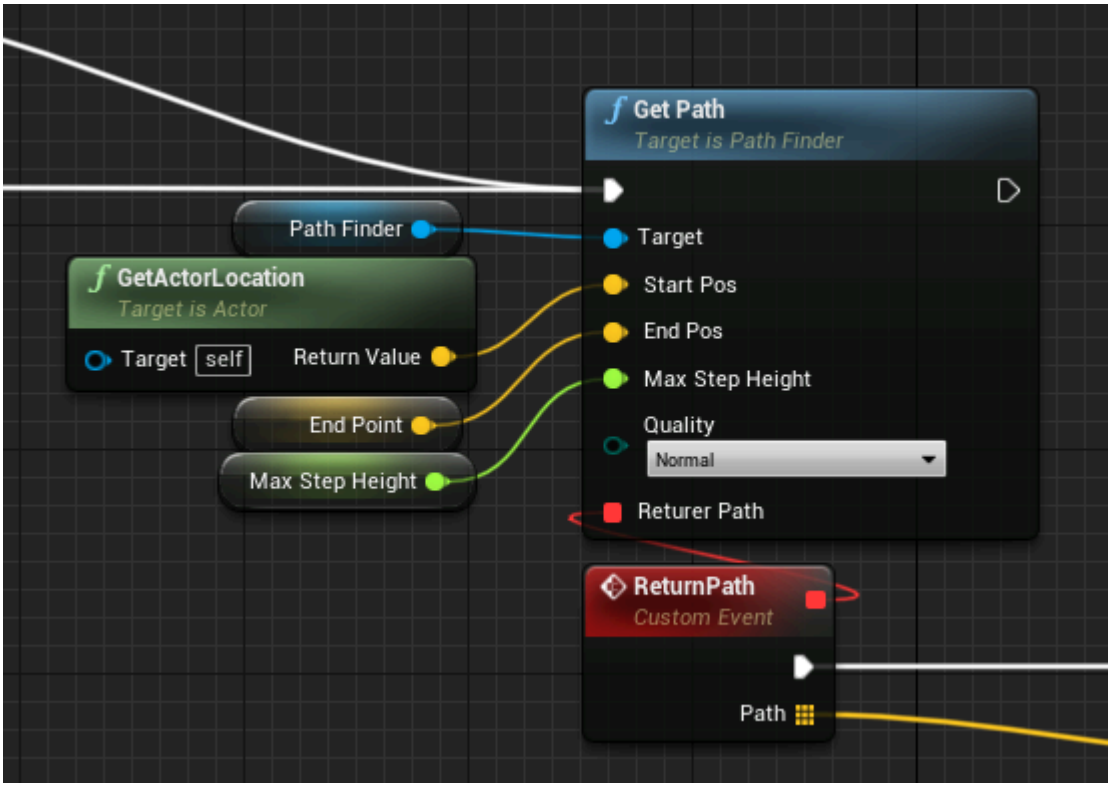


4. If you need to regenerate the grid during the game, then BP_A_PathFinder has a function for manually regenerating the grid. You need to call the Manual Regenerate Grid function



5. To get the path in the AI, you need to call the getPath function:





Input:

   a. Start Pos - location in the world from where you need to build a path.
   b. End Pos - location in the world, where you need to build a path.
   c. Max Step height - the maximum step height. If you have an object on the stage that a drone can fly into, for example,but a zombie cannot enter, then feel free to play with this value.
   d. Quality - The quality of the path search. it has three modes:
      - the first is the fastest, but less accurate;
      - the second one is a little longer, but almost always gives the right path;
      - the third method is the longest, but it gives almost a 100% chance of finding
   e. Returner Path - a delegate that will be called after building a path at the output of which your path

6. There is a primitive optimizer of the resulting path, which greatly reduces the volume of the path, but not its quality. To use it, you need to call a separate function

OptimizePath