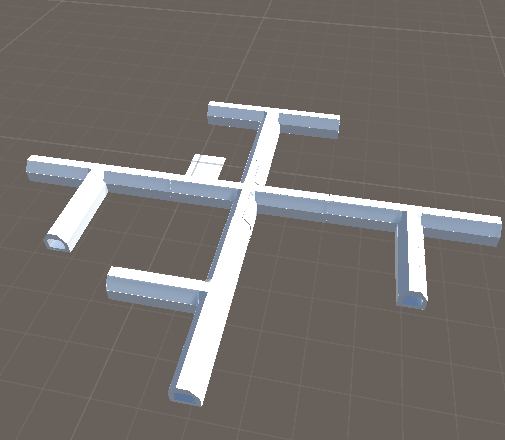
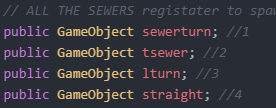
Documentation Sewer Generation.

# Pre-Made patterns.

There are Pre-Made patterns for the Generation to properly generate The Sewers with less collision as possible by using Vector3 coordinates(position) and Quaternion.Euler(rotation).

It will also help Check any position error That might happens in the test run of the coding. It also helps the code knows which sewer can connect to it and how the sewer connect to it since there are 4 different types of sewers.

# Register the sewers

This piece of coding register each sewer and set them to a number for many possible combination. For a example, Sewer Number 3 (L turn sewer) Might be build on sewer Number 2 (T Sewer).

# The Amount of sewers.

This count variable set the max amount of sewers patterns that can be generate.

# Make a sewer.

This make a sewer. The first agreement is what sewer is being made. Second argument is the position. The third agreement tell it rotation. It data is store in newsewer.

# Position of new sewer.

It save the newest made sewer position for other sewer to build on.

# Side of the sewer.

This will check each side of the sewer to see if there are a sewer that already been filled up

And it allow for no duplication sewer on one side only.

# 

# Multiple side sewer

It will Check if the sewer is going to have another sewer build on

# 

The range goes from 1 to 2. If it is 2 another sewer is built on. If it 1, then it picks the newest sewer for another pattern.

# 

# Sewer picks

 It picks the sewer number 1 to 4 to build atop of the last sewer.

The last sewer default is 1 and if the newest sewer is done, then the last sewer should be the number of the newest sewer.

# Looks

Most of the sewer generators is made out of a bunch of if statements and many different variables.

It is also well commented that people will know when I got left off and knows about each function will do.

