Wave Function Collapse

Research
https://www.youtube.com/watch?v=2SuvO4Gi7uY

- It's like sudoku!
- Superposition = occupying all spaces at once. When a number is filled in, all superpositions are collapsed into a single possibility.
- Propagate = remove options that we have info on
- After filling in everything given, look for the box with the least number of choices (lowest entropy and collapse it to a single possibility)
- Prioritize low entropy cells, since it minimizes the chance of making a bad choice

Takes a grid of cells with superpositions (called a wavefunction)

- Each cube has their own adjacency rules
- One iteration = cancel out all cells that are propagated from the selected tile

Adjacency Rules

- Socket system = numbers/letters that identify a connection type
    - Color matches at certain points
- Infinite world Demo

Steps

- First, create a list of 6 lists of valid neighbors (one for each cube face)
- Whenever we collapse a cell, we remove any modules from neighboring cells that aren't the list of valid neighbors
- Lable each module with sockets
- Check sockets and if they can fit together
- Loop over every module in the set and store the position of each vertex that sits along each edge of the 6 boundaries **[this is all scripted in Blender, do not manually do this]**
- Store and label these as we go
- This creates a dictionary of module names and profiles
- Special markup for symmetrical and asymmetrical: tag symmetrical with s
    - Symmetrical will always fit with itself
    - Asymmetrical will fit with a mirrored (stored as two different sockets and mark one with an F
- Special markup for Top and Bottom
    - Store 4 versions of each socket, labeling them with a rotation index
    - Verticle sockets will only be considered valid if they have the same name and socket index

Prototypes

- Metadata for modules containing info about which mesh to use, what the rotation is, and the 6 lists of valid neighbors
- This means you don't have to export 4 different meshes for each rotation, instead you reference the same mesh with a different rotation value
- Write to a JSON file, which you can load as a Dictionary in-game engine

For each module = 4 prototype entries containing mesh name, rotation index, and socket types.
Compare each prototype with every other prototype 6 times (once for each direction) check that the opposing sockets are valid using adjacency rules.
```
If symmetrical (0s):
        SocketA == SocketB
If asymmetrical (0, 0f):
        SocketA == SocketB + "f"
If vertical (v0_0)
        Socket A == SocketB && SocketA.index == SocketB.index
```
Add this to the valid neighbor list

If there are no vertices on a particular side of a module, this means the socket is stored as -1. After creating all prototypes, create a blank socket with a blank mesh (-1 on each side)

Basic Engine steps:

1. Load prototype data (maybe txt file from JSON) then convert is to a dictionary
2. Load wave function (create a 3-dimensional array, filling each cell with a copy of the prototype data)
3. Create a while loop that breaks if the wave function is fully collapsed, if not then iterate
    a. Iterate function finds the cell with the lowest entropy, with randomization if there is a tie
    b. Pick a prototype from

Topic Outline

- What is Wave From Collapse
- How can Wave From Collapse be used
- How does the Wave Form Collapse Algorithm work
- What can Wave Form Collapse be used for
- References

Demos

https://bolddunkley.itch.io/wfc-mixed
https://bolddunkley.itch.io/wave-function-collapse

http://oskarstalberg.com/game/wave/wave.html

https://marian42.de/article/wfc/
https://github.com/mxgmn/WaveFunctionCollapse
https://robertheaton.com/2018/12/17/wavefunction-collapse-algorithm/