**Assignment #4: A-Maze**

As Waldorf continues his exploration of the inter dimensions he has found himself stuck within a maze. Can you help him escape?

In this assignment, students must create a maze from XML data which Waldorf will be placed in. The students must implement basic states and path finding to help Waldorf escape from the maze without getting stuck!

**Objective:**

* Gain familiarity with XML data in a project
* Create a 3D world using tiles
* Expose students to basic behavior concepts and pathfinding

**Required Resources:**

* A-maze Unity Project (Unity 2018.2.3f1)

**Project Structure:**

* “TheMaze”: object with MazeBehaviour script
  + Generates all the tiles
  + Generates Waldorf
  + Positions the camera
  + Empty containers to generate objects within
* “maze\_camera”: main camera with CameraBehaviour to track Waldorf
  + No need to adjust
* Animations folder: all animations used by Waldorf
* Prefabs folder: the prefabs used to generate the world
  + “waldorf”: object with animations and WaldorfBehaviour
  + “tile\_open”: navigable tile with TileBehaviour
  + “tile\_wall”: blocking tile with TileBehaviour
* Data folder: the XML data used to generate levels
  + “maze\_flat”: a maze of all open tiles for testing
  + “maze\_easy”: an easy to navigate maze for basic testing
  + “maze\_hard”: a difficult to navigate maze
* Starting Scripts:
  + CameraBehaviour (DONE): moves camera with Waldorf
  + TileBehaviour (DONE): controls tile state and info about movement
  + MazeBehaviour: reads XML data to generate the maze
  + WaldorfBehaviour: controls the state, movement, and picks tiles to move to as targets for the Waldorf object

**The Components:**

* XML:
  + Must parse data from XML to generate the level
  + See comments in “maze\_easy” for format details
  + Objects should be created from prefabs
  + Must support “uneven” rows and columns
    - I WILL test with some rows having less columns than others (i.e. not just square grids)
* Waldorf
  + MUST have four states defined and indicated using animations
    - Think: a short period between each move where Waldorf “thinks” about where to go next (“think” animation)
      * Time must be given for this state which then picks a tile and begins to move
    - Move: a state where Waldorf moves from his tile to a target tile (“move” animation)
      * Based on the target tile and using the functions provided
    - Fall: a state where Waldorf falls/jumps off the map when he reaches a point with no tiles in a direction, the exit (“fall” animation)
      * If there is no tile in a direction the end has been found
      * MUST have some kind of falling done in any manner you choose
    - Stuck: a state where Waldorf has no valid tiles to move to thus ending the adventure (“stuck” animation)
      * No need for a reset
  + MUST pick the right tile to move towards
    - Each tile has a TileWeight (value between 0-1) which determines how valid a tile is 1=valid, 0=not valid
      * Found in TileBehaviour
    - Can only move to adjacent tiles with the highest weighted tile being selected
    - If there are multiple tiles with the same weight a random tile from these weights should be selected
    - Once selected the targeted tile should become lower weighted (implemented already in TileBehaviour)

**Submission:**

* Completed Unity Project
  + Only “Assets”, “Packages”, and “Project Settings” folders
  + Naming convention for project folder: gat240\_studentid\_4
  + Submit as zip: gat240\_studentid\_4.zip

**Rubrics:**

* **XML Parsing – 20%**
  + MUST parse data from XML
  + MUST handle changes to XML data
* **Maze Generation – 20%**
  + Use XML data to generate a maze using prefabs
  + Place Waldorf in correct location and set camera
  + Support for non-square grids (not all rows have same amount of tiles)
* **Object States – 20%**
  + Waldorf correctly changes between 4 different states
  + Tiles correctly change between states
* **Pathfinding – 20%**
  + Waldorf correctly picks tiles based on the given formula
  + Successful exit of maze
* **Custom Maze – 10%**
  + Inside of the data folder, a “maze\_custom” should be created
  + MUST be at least 20x20
  + MUST include at least 1 additional tile type (represented by a 2 in data)
    - Students should create this as a prefab “tile\_custom”
    - Pathfinding can take this new tile into account as student’s desire
* **Cleanliness – 10%**
  + Clean project structure
  + Clean code structure
  + Followed naming conventions
  + Explained complicated code

**Notes:**

Chapters 6 in the book can help a LOT in finding solutions. For XML parsing, there are many online resources.

Also, this project has some oddities with orientations causing the maze to be logically upside-down and the Z direction flipped. No need to fix this issue, just work around it. You are welcome to fix the problem if it is making things harder as long as motion is based on screen representation (X: left and right; Z: up[towards top of screen] and down).