GADE7321 POE Part 2 Documentation

“War in the Wastes”

Kerwin Naidoo ST10038448

Bradley Wicks ST10025342

Contents

GitHub Repository

https://github.com/Coolguybrad/GADE7321-Part-2

High Concept Statement

"War in the Wastes" is a board game that combines the strategy of chess and the excitement of the wasteland. Set in a post-apocalyptic world where resources are scarce, each piece represents a wasteland scavenger, each with its own specialties and strengths. Players must outmaneuver their competition and use each piece's unique strengths to gain the advantage and steal their resources. It's not only about taking your enemy's resources... it's about doing whatever it takes to survive the wastelands of *"War in the Wastes"*.

Rules

Objective: Either capture all the opponent's pieces or reach the goal on the opponent's side of the board (their resources).

The board: a unique plus shaped board with special tiles that have different effects.

Pieces: each side has 10 pieces, each with a different rank that affects who they can capture.

Movement: each piece has a standard move where they can move 1 space left right up or down; some pieces have a special quality that affects their movement.

Capturing: the player's piece can move onto the space an opponent's piece occupies to capture it if the opponent's character is of lower or equal rank. The rank of a piece is displayed above it and can be affected by certain tiles. Some characters have special qualities that affect who they can capture.

Pieces:

Rank 1: Sludge Mutant (can walk through toxic sludge, marked with an M)
Rank 2: Vulture
Rank 3: Scavenger
Rank 4: Brawler
Rank 5: Guard
Rank 6: Jumper (can jump across the toxic sludge, marked with a J)
Rank 7: Follower
Rank 8: Folk Hero

Special Tiles:

- High ground: while a character is on high ground it increases its rank by 1.

- Rough terrain: while a character is on rough terrain it decreases its rank by 1.
- Toxic sludge: Some characters can hide in the sludge, increasing their power and allowing them to capture any adjacent piece.
- Traps: Characters on a trap will be able to be captured by any piece no matter of rank. These tiles lie adjacent to the goal tile.

<u>Special qualities:</u>

- Hide in toxic sludge.
    - Characters who can hide in sludge tiles can kill any piece from that sludge.
- Jump over toxic sludge.
    - Cannot jump if any piece blocks the tile the character would jump to, or if the adjacent sludge tile is occupied.

Game State Representation:

Board:
The board is made up of an array of tile prefabs. These tile prefabs have child objects for a highlight that allows the player to see what their cursor is pointing at and a game object that shows up when a piece can move to that tile. Tiles have a tile script which stores the child game objects mentioned, whether the tile is occupied by a piece, what the piece occupying the tile is, an enum storing its tile type, and its coordinates.
The board is made up of different tile types being:
Normal: These are regular tiles with no special properties.
High ground: These tiles are raised from the board and increase a piece's rank by 1.
Toxic Sludge: These tiles only allow the lowest ranking pieces to move through them.
Rough: These tiles decrease a piece's rank by 1
Trap: These tiles reduce a piece's rank to 0 meaning they can be captured by any piece and cannot capture any piece
Blue Goal: When a blue piece lands on this tile the blue team wins
Red Goal: When a red piece lands on this tile the red team wins

Board Manager:
An empty game object containing the BoardManager script manages the board and the tiles that make it up. The BoardManager stores an array of Tiles, a single Tile that is the most recently clicked Tile, and a reference to the blue team's goal Tile and one for the red team's goal Tile.
BoardManager shows the possible moves a piece can take when it is selected, it also wipes the board of all possible moves when nothing is selected or when another piece is selected before showing where that newly selected piece can move.
BoardManager also handles resetting the board and quitting the game.

Pieces:
Pieces store many different variables such as: Locations, if they can walk on sludge tiles, if they can jump over sludge tiles, what team they're on, their initial power, the power they are currently, a TMP_Text that displays the piece's power, the tile they are on and references to the BoardManager, PieceManager, and TurnHandler.

Piece Manager:
An empty game object with a script called PieceManager, this script stores the piece that is currently selected, an array of the blue team's pieces, an array of the red team's pieces, as well as a reference to the BoardManager and the TurnHandler.
The piece manager handles moving the selected piece and checking whether it can do any special movements based on the tiles around that it is able to move onto.

Turn Handling:

An empty game object named TurnHandler manages which team’s turn it is, it stores an integer which represents which team is playing, either being 0 or 1 for blue and red team respectively. TurnHandler also stores a TMP_Text which it can then edit to show which team’s turn it is currently.

Game State Utility Function

A piece will first check its possible moves and prioritize capturing a piece, the piece will check if it is threatened by an opponent's piece if it were to capture the piece, if it is threatened, that path will be filtered out and a score that is calculated will run to find what is the next best option, if that option also threatens the piece it will check the next option until the best option that doesn't threaten the piece is found. Each piece's best possible move will be compared using a score that is calculated using the below equation, the piece with the highest scoring best move will make that move on the AI's turn.

Key:
mat – material
mobil – mobility
r – resource protection
d – distance
rank – piece rank
lm = Legal move count
pIR = pieces in radius of goal
rankAI = rank of AI piece
rankP = rank of player piece

$$Score = mat + mobi + r - d + rank$$

Material value is calculated by comparing the sum of differences between each side's pieces of equal ranks.

$$mat = [1(M - M') + 2(V - V') + 3(S - S') + 4(B - B') + 5(G - G') + 6(J - J') + 7(F - F') + 8(H - H')]$$

Mobility value is the total legal moves a piece can perform.

$mobil = lm$

Resource protection value is the difference between the amount of enemy pieces and own pieces in a 7x3 area around the goal.

$$r = (pIR' - pIR)$$

Distance is the distance between a piece and the goal.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Rank is the difference between the piece's rank and the opposition's piece's rank.

$$rank = rankAI - rankP$$