STACKER CLASS IN-DEPTH DOCUMENTATION

OVERVIEW

The `Stacker` class is a TypeScript implementation designed for the Treasure Hunter Challenge. It encapsulates the logic required for an agent to navigate a map, identify obstacles, pick up and stack blocks, and ultimately reach the treasure.

CLASS STRUCTURE

- **Constants:** Represents different cell types (EMPTY, WALL, BLOCK, GOLD).
- **State Variables:** Includes flags for carrying a block, current level, current position, and gold position.

KEY METHODS

UTILITY FUNCTIONS

- `isOneLevelAboveOrBelow(cellA, cellB)` : Determines if two cells are at adjacent levels.
- `isSameLevel(cellA, cellB)` : Checks if two cells are on the same level.

PATHFINDING

- `findPathToTarget(start, goal, gameMap)` : Implements the A* algorithm for pathfinding.

VISIBILITY AND POSITION UPDATES

- `canSeeGoalFrom(currentPos, gameMap)` : Checks if the goal is visible from the current position.
- `updateGoldPosition(currentPos, gameMap)` : Updates the agent's knowledge of the gold's position.

DECISION MAKING

- `decideNextMove(cell, gameMap)` : Determines the next move based on the current state and map.

PATH TO DROP LOCATION

- `findPathToDropLocation(currentPos, goldPos, gameMap)` : Finds the path to the best drop location for a block.

MOVE TRANSLATION

- `getNextMoveFromPath(pathToTarget, currentPos)` : Translates the next step in a path to a move action.

BLOCK HANDLING

- `shouldPickupBlock(cell)` : Determines if the agent should pick up a block.
- `explore(gameMap, currentPos)` : Explores the map to find blocks or the gold.

TURN FUNCTION

- `turn(cell, gameMap)` : The main function called every game cycle to decide the agent's action.

STARTING POSITION INFERENCE

- `inferStartingPosition(cell, gameMap)` : Infers the starting position of the agent.
- `matchesStartingPattern(x, y, gameMap, cell)` : Checks if a cell matches the starting pattern.

CONCLUSION

The `Stacker` class is a comprehensive solution for the Treasure Hunter Challenge, demonstrating efficient pathfinding, strategic decision-making, and effective handling of game mechanics.