Musical Chairs – Java

A game of musical chairs where players compete to find an available chair when the music stops. The Emcee class serves as the game host, the Manager class manages chairs and game state, and the Player class represents the game participants. The Chair class represents the chairs used in the game.

This project demonstrates the use of synchronization and thread communication to simulate a musical chairs game. Players attempt to acquire chairs when the music stops, and the game progresses through multiple rounds until one player wins.

Chair Class

Description:

The `Chair` class represents a chair object used in a game of musical chairs. It contains information about the chair's name, occupancy status, and the player currently occupying the chair.

Attributes:

- `name` (String): The name of the chair.
- `occupied` (boolean, volatile): A boolean indicating whether the chair is occupied by a player.
- `occupiedBy` (Player, volatile): A reference to the player currently occupying the chair.

Constructors:

- `Chair(String name)`: Creates a new chair object with the specified name. Initially, the chair is unoccupied.

Methods:

- `acquireChair(Player p)`: A synchronized method used by a player to attempt to sit on the chair. If the chair is unoccupied, the player occupies it and returns `true`. If the chair is already occupied, the method returns `false`.
- `toString()`: Overrides the `toString()` method to provide a string representation of the chair, which is its name.

Emcee Class

Description:

The `Emcee` class represents the host of the musical chairs game. It manages the game rounds, players, and interactions between players and chairs.

Attributes:

- `numPlayers` (int): The total number of players in the game.
- `players` (ArrayList of Player): A list of Player objects representing the participants.
- `manager` (Manager): An instance of the Manager class to manage chair and game state.
- `currentRound` (int): The current round number.
- `totalRounds` (int): The total number of rounds required.

Constructors:

- `Emcee(int n)`: Creates an Emcee object with the specified number of players (n). It initializes the manager, player list, and round information.

Methods:

- `initPlayers()`: Initializes the player objects, associating each player with the manager object for communication.
- `removePlayer(String name)`: A synchronized method to remove a player from the list of active players based on their name.
- `run()`: Overrides the `run()` method for the Emcee thread. It controls the game rounds, starting the players, managing music, declaring losers, and finally declaring the winner.

Manager Class

Description:

The `Manager` class is responsible for managing the chairs, turning the music on/off, and preparing chairs for the next round in the musical chairs game.

Attributes:

- `chairs` (ArrayList of Chair): A list of Chair objects representing the available chairs.
- `roundWinner` (String): Stores the name of the winner of the current round.
- `roundLoser` (String): Stores the name of the player who loses in the current round.
- `numPlayers` (int): The number of active players in the game.

Constructors:

- `Manager(int numPlayers)`: Creates a Manager object with the specified number of players (numPlayers).

Methods:

- `initChairs()`: Initializes the list of chairs, creating Chair objects for each player.
- `circleChairs()`: A synchronized method that makes player threads wait until music is turned off.
- `musicOff()`: A synchronized method to turn off the music and notify all player threads to start looking for chairs.
- `prepareChairsForNextRound()`: A synchronized method that removes the last chair and resets all other chairs for the next round.

Player Class

Description:

The `Player` class represents a player participating in the musical chairs game. It interacts with chairs and communicates with the manager.

Attributes:

- `name` (String): The name of the player.
- `manager` (Manager): A reference to the Manager object for communication.
- `alive` (boolean): Indicates whether the player is still in the game.
- `cont` (boolean): A flag used to continue to the next round.

Constructors:

- `Player(String name, Manager manager)`: Creates a Player object with a name and a reference to the game manager.

Methods:

- `findChair()`: Randomly selects a chair to sit on, shuffling the order of chairs to increase randomness.
- `run()`: Overrides the `run()` method for the Player thread. It waits for the music to stop, looks for a chair, records winner and loser information, and prepares for the next round.