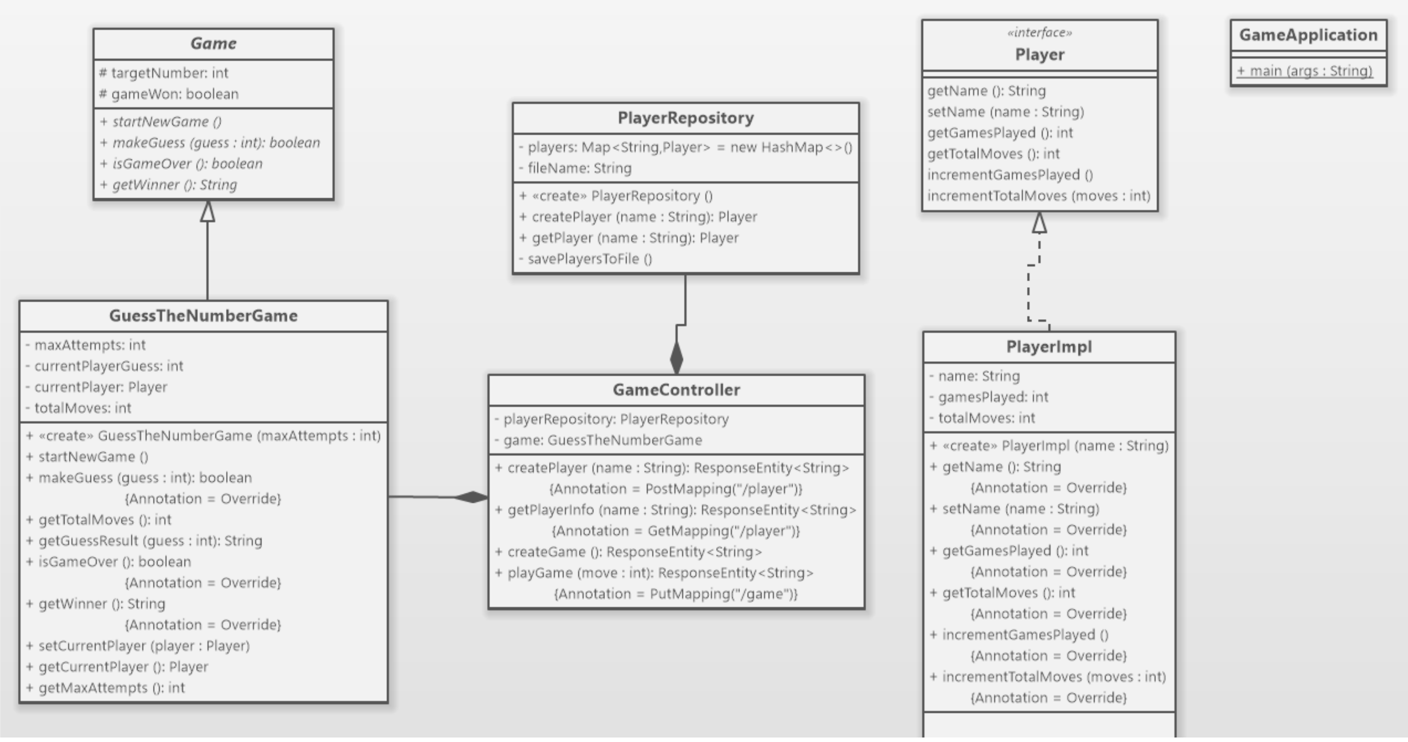
Design Document

Introduction

The "Guess the Number" game is a Java-based application that allows players to guess a random number within a specified range. This design document outlines the architecture, Java classes, and their interactions within the application.


Game
# targetNumber: int
# gameWon: boolean
+ startNewGame ()
+ makeGuess (guess : int): boolean
+ isGameOver (): boolean
+ getWinner (): String
PlayerRepository
- players: Map<String,Player> = new HashMap<>()
- fileName: String
+ «create» PlayerRepository ()
+ createPlayer (name : String): Player
+ getPlayer (name : String): Player
- savePlayersToFile ()
«interface»
Player
getName (): String
setName (name : String)
getGamesPlayed (): int
getTotalMoves (): int
incrementGamesPlayed ()
incrementTotalMoves (moves : int)
GameApplication
+ main (args : String)
GuessTheNumberGame
- maxAttempts: int
- currentPlayerGuess: int
- currentPlayer: Player
- totalMoves: int
+ «create» GuessTheNumberGame (maxAttempts : int)
+ startNewGame ()
+ makeGuess (guess : int): boolean
{Annotation = Override}
+ getTotalMoves (): int
+ getGuessResult (guess : int): String
+ isGameOver (): boolean
{Annotation = Override}
+ getWinner (): String
{Annotation = Override}
+ setCurrentPlayer (player : Player)
+ getCurrentPlayer (): Player
+ getMaxAttempts (): int
GameController
- playerRepository: PlayerRepository
- game: GuessTheNumberGame
+ createPlayer (name : String): ResponseEntity<String>
{Annotation = PostMapping("/player")}
+ getPlayerInfo (name : String): ResponseEntity<String>
{Annotation = GetMapping("/player")}
+ createGame (): ResponseEntity<String>
+ playGame (move : int): ResponseEntity<String>
{Annotation = PutMapping("/game")}
PlayerImpl
- name: String
- gamesPlayed: int
- totalMoves: int
+ «create» PlayerImpl (name : String)
+ getName (): String
{Annotation = Override}
+ setName (name : String)
{Annotation = Override}
+ getGamesPlayed (): int
{Annotation = Override}
+ getTotalMoves (): int
{Annotation = Override}
+ incrementGamesPlayed ()
{Annotation = Override}
+ incrementTotalMoves (moves : int)
{Annotation = Override}

Architecture

The application follows a simple client-server architecture where the server hosts the game logic, and clients can interact with the game via HTTP requests. The game consists of players, games, and game moves.

Java Classes

Player.java

- **Description**: Represents a player in the game.
- **Methods**:
  - `getName()` : Get the player's name.
  - `setName(name)` : Set the player's name.
  - `getGamesPlayed()` : Get the number of games played by the player.
  - `getTotalMoves()` : Get the total moves made by the player.
  - `incrementGamesPlayed()` : Increment the number of games played by the player.
  - `incrementTotalMoves(moves)` : Increment the total moves made by the player.

PlayerImpl.java

- **Description**: An implementation of the `Player` interface.
- **Methods**: Inherits methods from the `Player` interface.

GuessTheNumberGame

- **Description**: Represents the guessing game.

- **Fields**:

  - `maxAttempts` : The maximum number of attempts allowed.
  - `targetNumber` : The random number to be guessed.
  - `currentPlayerGuess` : The current guess made by the player.
  - `currentPlayer` : The current player.
  - `gameWon` : A flag to indicate if the game has been won.
  - `totalMoves` : The total number of moves made in the game.

- **Methods**:

  - `startNewGame()` : Start a new game.
  - `makeGuess(guess)` : Make a guess in the game.
  - `isGameOver()` : Check if the game is over.
  - `getWinner()` : Get the winner of the game.
  - `getGuessResult(guess)` : Get the result of a guess (e.g., "The number is larger," "The number is smaller").

PlayerRepository.java

- **Description**: Manages players and their data.
- **Fields**:
  - `players` : A map of player names to player objects.
  - `fileName` : The name of the file to save player data.
- **Methods**:
  - `createPlayer(name)` : Create a new player.
  - `getPlayer(name)` : Get a player by name.
  - `savePlayersToFile()` : Save player data to a text file.

GameController.java

- **Description**: Handles HTTP requests and game interactions.
- **Fields**:
  - `playerRepository` : An instance of the `PlayerRepository` class.
  - `game` : An instance of the `GuessTheNumberGame` class.
- **Endpoints**:
  - `/player` : Create a player and get player information.
  - `/game` : Create a game and make moves in the game.

Class Interactions

- The `Player` and `PlayerImpl` classes handle player-related data and statistics.
- The `GuessTheNumberGame` class manages the game logic, including starting new games, making guesses, and determining game outcomes.
- The `PlayerRepository` class manages player data and file I/O.
- The `GameController` class handles HTTP requests, interacts with the `PlayerRepository` , and controls game flow.

API Endpoints

Base URL: `http://localhost:8080`

Player API

Create a Player

- **URL:** POST `/player`
- **Parameters:** `name` (string) - The name of the player.
- **Demo Call:** `curl -X POST "http://localhost:8080/player?name=Shovan"`

Get Player Information

- **URL:** GET `/player`
- **Parameters:** `name` (string) - The name of the player.
- **Demo Call:** `curl "http://localhost:8080/player?name=Shovan"`

Game API

Create a Game

- **URL:** POST `/game`
- **Parameters:** `playerName` (string) - The name of the player who starts the game.
- **Demo Call:** `curl -X POST "http://localhost:8080/game?playerName=Shovan"`

Make a Move in the Game

- **URL:** PUT `/game`
- **Parameters:** `move` (int) - The number guessed by the player (0-99).

- **Demo Calls:**
    - `curl -X PUT "http://localhost:8080/game?move=5"`
    - `curl -X PUT "http://localhost:8080/game?move=8"`
    - `curl -X PUT "http://localhost:8080/game?move=6"`

Get Game Moves

- **URL:** GET `/game/moves`
- **Demo Call:** `curl "http://localhost:8080/game/moves"`

Player Information After the Game

- **URL:** GET `/player`
- **Parameters:** `name` (string) - The name of the player.
- **Demo Call:** `curl "http://localhost:8080/player?name=Shovan"`

Notes

- Replace `"http://localhost:8080"` with the appropriate base URL if you are hosting the API on a different server or port.
- You can create players and start games before making moves.
- Game results will be reflected in player statistics.

Enjoy playing the Number Guessing Game!