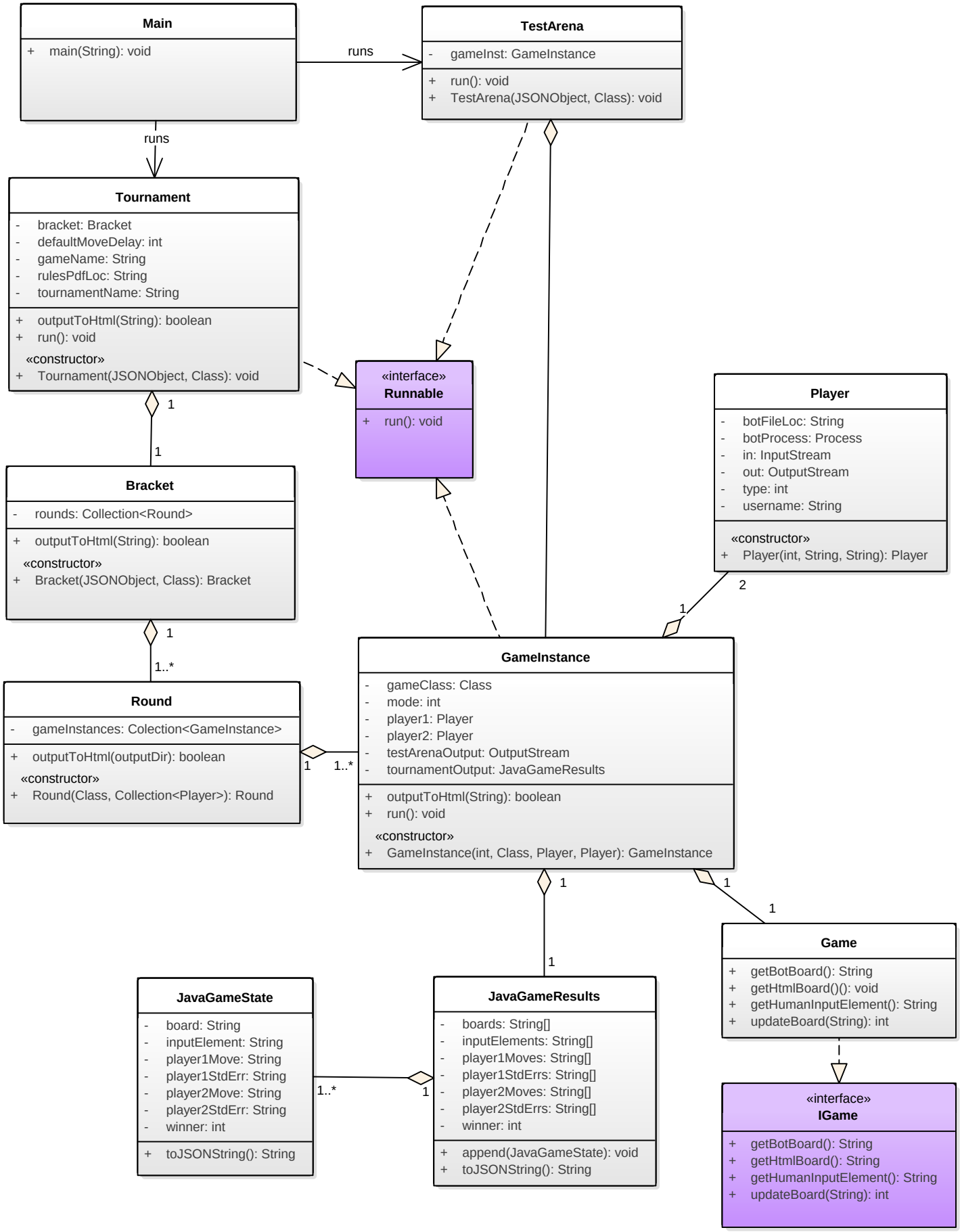
Main
+ main(String): void
runs
TestArena
- gameInst: GameInstance
+ run(): void
+ TestArena(JSONObject, Class): void
runs
Tournament
- bracket: Bracket
- defaultMoveDelay: int
- gameName: String
- rulesPdfLoc: String
- tournamentName: String
+ outputToHtml(String): boolean
+ run(): void
«constructor»
+ Tournament(JSONObject, Class): void
«interface»
Runnable
+ run(): void
1
1
Bracket
- rounds: Collection<Round>
+ outputToHtml(String): boolean
«constructor»
+ Bracket(JSONObject, Class): Bracket
Player
- botFileLoc: String
- botProcess: Process
- in: InputStream
- out: OutputStream
- type: int
- username: String
«constructor»
+ Player(int, String, String): Player
2
1
1
1..*
Round
- gameInstances: Colection<GameInstance>
+ outputToHtml(outputDir): boolean
«constructor»
+ Round(Class, Collection<Player>): Round
1
1..*
GameInstance
- gameClass: Class
- mode: int
- player1: Player
- player2: Player
- testArenaOutput: OutputStream
- tournamentOutput: JavaGameResults
+ outputToHtml(String): boolean
+ run(): void
«constructor»
+ GameInstance(int, Class, Player, Player): GameInstance
1
1
1
1
Game
+ getBotBoard(): String
+ getHtmlBoard()(): void
+ getHumanInputElement(): String
+ updateBoard(String): int
JavaGameState
- board: String
- inputElement: String
- player1Move: String
- player1StdErr: String
- player2Move: String
- player2StdErr: String
- winner: int
+ toJSONString(): String
1..*
1
JavaGameResults
- boards: String[]
- inputElements: String[]
- player1Moves: String[]
- player1StdErrs: String[]
- player2Moves: String[]
- player2StdErrs: String[]
- winner: int
+ append(JavaGameState): void
+ toJSONString(): String
«interface»
IGame
+ getBotBoard(): String
+ getHtmlBoard(): String
+ getHumanInputElement(): String
+ updateBoard(String): int