# HTW View

**« Interface »**
**MazeGUI**

- getMazeSize(): Integer
- getPlayers(): Players[]
- closestDistanceFromWumpus(Integer): Integer
- closestDistanceFromPit(Integer): Integer
- closestDistanceFromWumpus(): Integer
- getCells(): Cell[][]
- getCurrentPlayer(): Integer
- getRestrictedPlayer(): Integer

**« Interface »**
**Maze**

- shootArrow(GlobalEnum.direction, Integer): String
- move(GlobalEnum.direction): String
- showStatus(): String
- closestDistanceFromPit(): Integer
- closestDistanceFromWumpus(): Integer
- print(): String
- setRestrictedPlayer()
- switchPlayer()

**MazeImpl**

- wumpusPosition: Point
- numberOfWallsRemoved: Integer
- allEdjes: List<Edge>
- mapParentCount: HashMap<String, Integer>
- cells: Cell[][]
- parent: String[][]
- totalBats: Integer
- totalPits: Integer
- player: Players[]
- totalArrows: Integer
- remainingWalls: Integer
- columnSize: Integer
- rowSize: Integer
- startPosition: Point
- mapDirectionToValue: HashMap<GlobalEnum.direction,int[]>
- mapWallsToIndex: HashMap<GlobalEnum.direction,Integer>
- updateCoordinate: UpdateCoordinate
- matchStatus: GlobalEnum.status
- matchWonMsg: String
- matchLostMsg: String
- arrowDistanceMsg: String
- winnerMsg: String
- tunnelMsg: String
- caveMsg: String
- playerPositionMsg: String
- fellPitMsg: String
- batCapturedMsg: String
- wumpusCaughtMsg: String
- noTunnelsMsg: String
- noCavesMsg: String
- noBatMsg: String
- noArrowMsg: String
- noPitMsg: String
- noUpdateCoordinateMsg: String
- fillWumpus()
- fillBatsAndPits()
- generateMaze()
- generateAllEdjes()
- getRandomInteger(Integer): Integer
- removeWalls(Integer, Integer, Integer, Integer)
- union(Integer, Integer, Integer, Integer)
- find(String): String
- removeExtraEdjes()
- removeCellWall(Integer, Integer, Integer, Integer, Integer)
- addTunnels()
- moveArrowThroughTunnel(Set<String>, Integer, Integer): int[]
- throwArrowsInCaveHelper(GlobalEnum.direction, Integer, Integer, Integer, Integer)
- fillWumpusDistance()
- fillPitDistance()
- findTunnels()
- findCaves(): String
- currentPosition(): String
- moveThroughTunnel(Integer, Integer, Set<String>)
- verifyObstacles(): String
- batCapturedHelper(): String
- getCells(): Cell[][]
- fillGoldAndTheif()

**« Interface »**
**UpdateCoordinate**

- updateValues(Integer, Integer, Integer, Integer): int[]

**SameCoordinate**

**NewCoordinate**

**Edge**

- PointB: Point
- pointB: Point
- getPointA()
- getPointB()
- setPointA()
- setPointB()

**Point**

- x: undefined
- y: undefined
- getX(): Integer
- getY(): Integer
- setX(Integer)
- setY(Integer)

**Player**

- currentLocation: Point
- getCurrentLocation(): Point
- setCurrentLocationX(Integer)
- setCurrentLocationY(Integer)
- setCurrentLocation(Point)
- setTotalGold(Integer)
- getTotalGold(): Integer
- getTotalArrow(): Integer

**Cell**

- distanceFromWumpus: Integer
- distanceFromPit: Integer
- isTunnel: Boolean
- isBat: Boolean
- isPit: Boolean
- wumpus: Boolean
- walls: boolean[]
- thief: Boolean
- gold: Integer
- setWallsByIndex(Integer, Boolean)
- getWallsIndexValue(Integer)
- getWalls(): boolean[]
- isTunnel(): Boolean
- setTunnel(Boolean)
- isBat(): Boolean
- setBat(Boolean)
- isPit(): Boolean
- setPit(Boolean)
- isWumpus(): Boolean
- setWumpus(Boolean)
- getDistanceFromWumpus(): Integer
- setDistanceFromWumpus(Integer)
- getDistanceFromPit(): Integer
- setDistanceFromPit(Integer)
- isTheif(): Boolean
- getWalls2(): Integer
- getGold(): Integer
- setGold(Integer)
- setThief(Boolean)
- setWallsByIndex2(Integer, Boolean)
- getWallsIndexValue2(Integer): Boolean

**NonWrappingMaze**

**WrappingMaze**

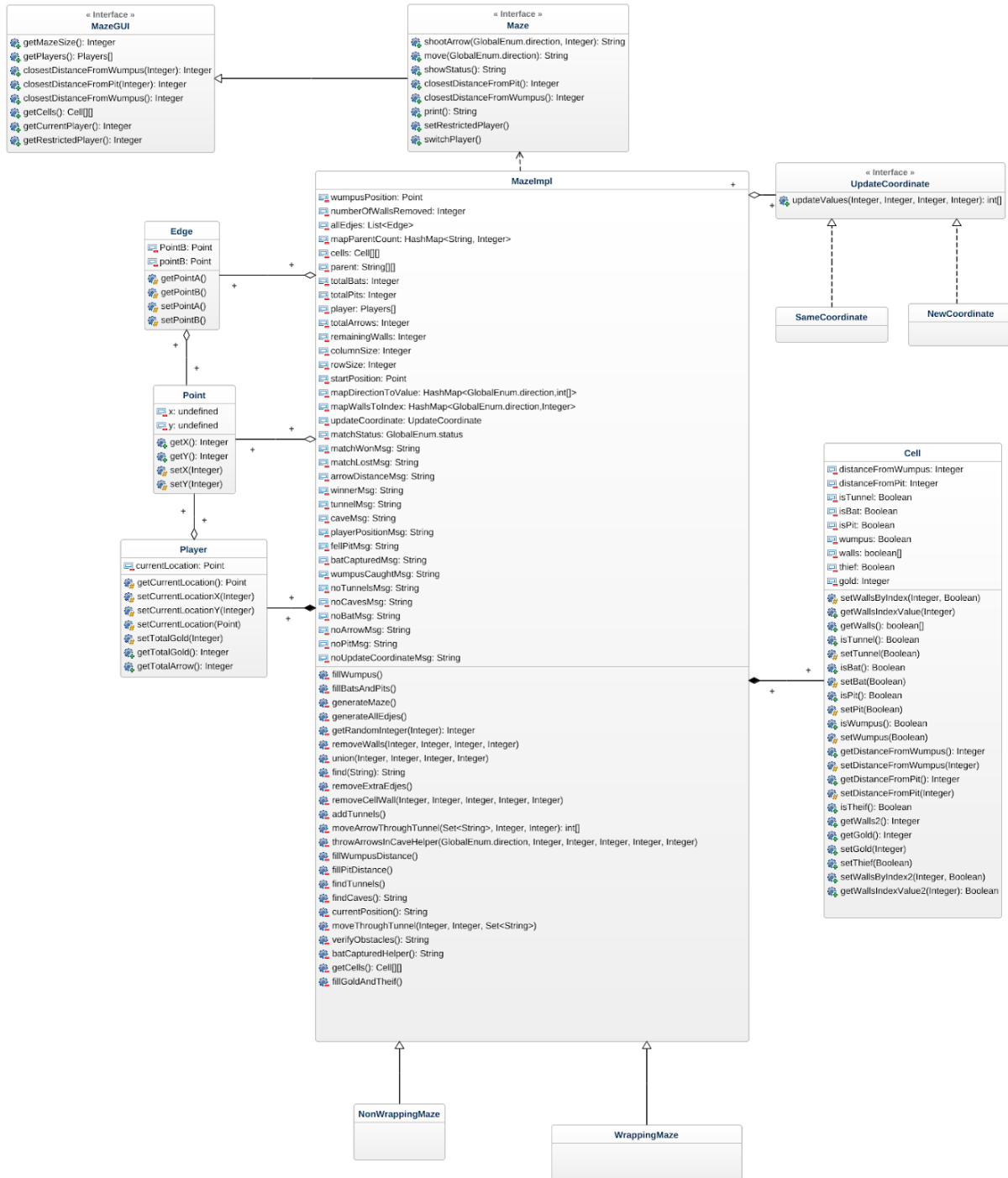## Explanation

In this design, we extended the wumpus game by adding functionality for two players. For this new functionality, we have an array of Players in MazeImpl to represent multiple players, and we also added gold and thief in the Cell for which we have a function fillGoldAndTheif to cells with it in MazeImpl class. We have also created a read-only implementation of the maze by creating a new interface MazeGUI which can be used by a user or a view for read-only purposes. We have also added a switchPlayer function which will be used to switch playing chances between the players.