Old UML Diagram:

Coordinate
- # UPPER_LEFT: String
- # UPPER_CENTER: String
- # UPPER_RIGHT: String
- # MIDDLE_LEFT: String
- # MIDDLE_CENTER: String
- # MIDDLE_RIGHT: String
- # LOWER_LEFT: String
- # LOWER_CENTER: String
- # LOWER_RIGHT: String

Player
- - name: String
- - player_type: char
- - squares: ArrayList<Coordinate>
- - turn: boolean
- <<Create>>Player(String name, boolean turn,, char player_type)
- + getName(): String
- + setName(String name)
- + getPlayerType(): char
- + setPlayerType(char player_type)
- + getTurn(): boolean
- + setTurn(boolean turn)
- + getSquares(): ArrayList<Coordinate>
- + setSquares(ArrayList<Coordinate>)

Game
- - grid: Grid
- - playerX: Player
- - playerO: Player
- - turn: boolean
- <<Create>>Game(Player playerX, Player, playerO,
- + checkWinConditionX(): boolean
- + checkWinConditionO(): boolean
- + endGame()
- + handleTurn()
- + handleInput()
- + play()

Grid
- - HEIGHT: int
- - WIDTH: int
- - grid: Square[][]
- <<Create>>Grid()
- + isFull(): boolean
- + toString(): String

Square
- - coord: Coordinate
- <<Create>>Square(Coordinate coord)
- + getCoordinate(): Coordinate
- + setCoordinate(Coordinate coord)
- + toString(): String
- + checkOverlap(): boolean

composition

Human
- <<Create>>Human(String name)

AI
- <<Create>> AI(String name, boolean turn)

uses

implements

Main

Empty
- <<Create>>Empty(Coordinat coord)
- + toString(): String

CellX
- <<Create>>CellX(Coordinate coord)
- + toString(): String

CellO
- <<Create>>CellO(Coordinate coord)
- + toString(): String

New UML Diagram:

Type
- + X: String
- + O: String

Coordinate
- # UPPER_LEFT: String
- # UPPER_CENTER: String
- # UPPER_RIGHT: String
- # MIDDLE_LEFT: String
- # MIDDLE_CENTER: String
- # MIDDLE_RIGHT: String
- # LOWER_LEFT: String
- # LOWER_CENTER: String
- # LOWER_RIGHT: String

Grid
- - HEIGHT: int
- - WIDTH: int
- - grid: Square[][]
- <<Create>>Grid()
- <<Create>>Grid(Grid other)
- + getSquare(): Square
- + setSquare(Square square)
- + getGrid(): Grid
- + setGrid(Grid)
- + isFull(): boolean
- + toString(): String

Player
- - name: String
- - player_type: char
- <<Create>>Player(String name)
- + getName(): String
- + setName(String name)
- + getPlayerType(): char
- + setPlayerType(char player_type)
- + play()

Game
- - SCANNER: Scanner
- - grid: Grid
- - playerX: Player
- - playerO: Player
- - currentPlayer: Player
- - isOver: boolean
- - winner: Player
- <<Create>>Game()
- + getGrid(): Grid
- + startGame()
- + checkWinConditionX(): boolean
- + checkWinConditionO(): boolean
- + handleTurn()
- + handleInput()
- + play()

Square
- - x: int
- - y: int
- <<Create>>Square(int x, int y)
- + setSquare(Square square)
- + getX(): int
- + getY(): int
- + setX(): int
- + setY(): int
- + toString(): String
- + checkOverlap(): boolean

Human
- <<Create>>Human(String name)
- + play(): String

AI
- - game: Game
- <<Create>> AI(String name, Game game)
- - minimax(Grid grid, int depth, boolean isMaximizing): int
- + play(): String
- + minimaxMove(): String

Main

Empty
- <<Create>>Empty(int x, int y)
- + toString(): String

CellX
- <<Create>>CellX(int x, int y)
- + toString(): String

CellO
- <<Create>>CellO(int x, int y)
- + toString(): String
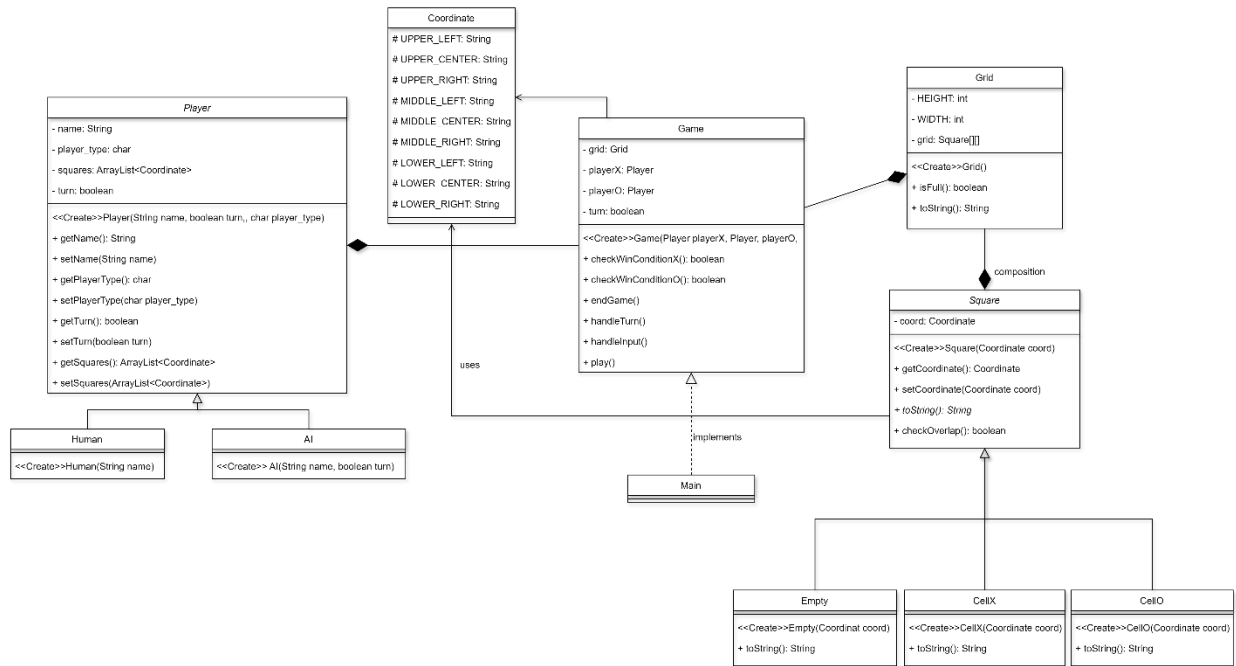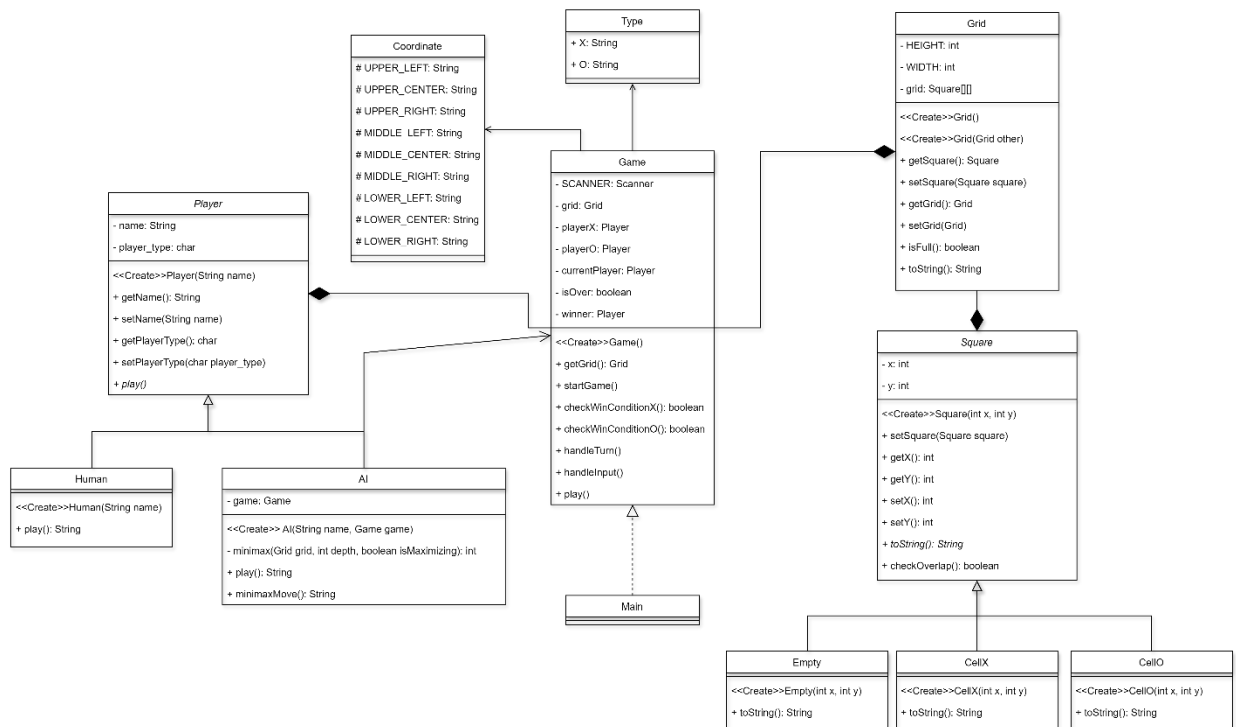
Updates:

1. I changed from the coordinate unit (e.g., "A1", "C3") to an x-y coordinate unit. That seemed easier to handle since the grid will use them and it's simpler. We treat the coordinate input when we receive to work with the x-y coordinates to reduce the complexity of the solution.

2. Deletion of the ArrayList to each player to save their coordinates. We treat the winning scenario in the Game class.

3. Transfer the turn marker to the Game class instead of the players since it's a game attribute.

4. Addition of a Type class to save the X and O along with their respective colors for clarity and visibility.

5. Addition of the missed play function for the Player, Human or AI. Addition of the Minimax algorithm for the computer to be smart and compete for the win instead of a random value.

6. Change of the constructor for class Game by removing its parameters and creating a new initialization method startGame() to make it more object oriented and for clarity.

7. Addition of a Scanner attribute to the Game after encountering issues with the input handling function.

GitHub Repository:

https://github.com/eliascharbelsalameh/XO_game