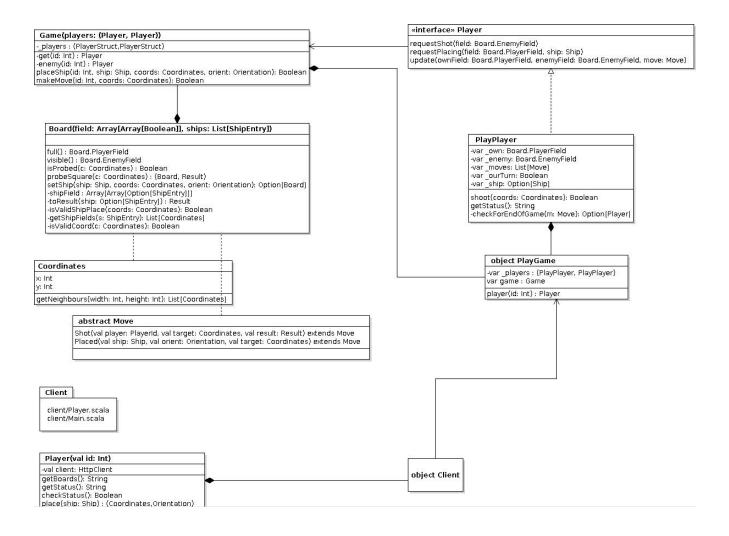This file was generated with Violet UML Editor 2.0.1.   (View Source / Download Violet)

Game(players: (Player, Player))

-_players : (PlayerStruct,PlayerStruct)

-get(id: Int) : Player
-enemy(id: Int) : Player
placeShip(id: Int, ship: Ship, coords: Coordinates, orient: Orientation): Boolean
makeMove(id: Int, coords: Coordinates): Boolean

«interface» Player

requestShot(field: Board.EnemyField)
requestPlacing(field: Board.PlayerField, ship: Ship)
update(ownField: Board.PlayerField, enemyField: Board.EnemyField, move: Move)

Board(field: Array[Array[Boolean]], ships: List[ShipEntry])

full() : Board.PlayerField
visible() : Board.EnemyField
isProbed(c: Coordinates) : Boolean
probeSquare(c: Coordinates) : (Board, Result)
setShip(ship: Ship, coords: Coordinates, orient: Orientation): Option[Board]
-shipField : Array[Array[Option[ShipEntry]]]
-toResult(ship: Option[ShipEntry]) : Result
-isValidShipPlace(coords: Coordinates): Boolean
-getShipFields(s: ShipEntry): List[Coordinates]
-isValidCoord(c: Coordinates): Boolean

PlayPlayer

-var _own: Board.PlayerField
-var _enemy: Board.EnemyField
-var _moves: List[Move]
-var _ourTurn: Boolean
-var _ship: Option[Ship]

shoot(coords: Coordinates): Boolean
getStatus(): String
-checkForEndOfGame(m: Move): Option[Player]

Coordinates

x: Int
y: Int

getNeighbours(width: Int, height: Int): List[Coordinates]

object PlayGame

-var _players : (PlayPlayer, PlayPlayer)
var game : Game

player(id: Int) : Player

abstract Move

Shot(val player: PlayerId, val target: Coordinates, val result: Result) extends Move
Placed(val ship: Ship, val orient: Orientation, val target: Coordinates) extends Move

Client

client/Player.scala
client/Main.scala

object Client

Player(val id: Int)

-val client: HttpClient

getBoards(): String
getStatus(): String
checkStatus(): Boolean
place(ship: Ship) : (Coordinates,Orientation)