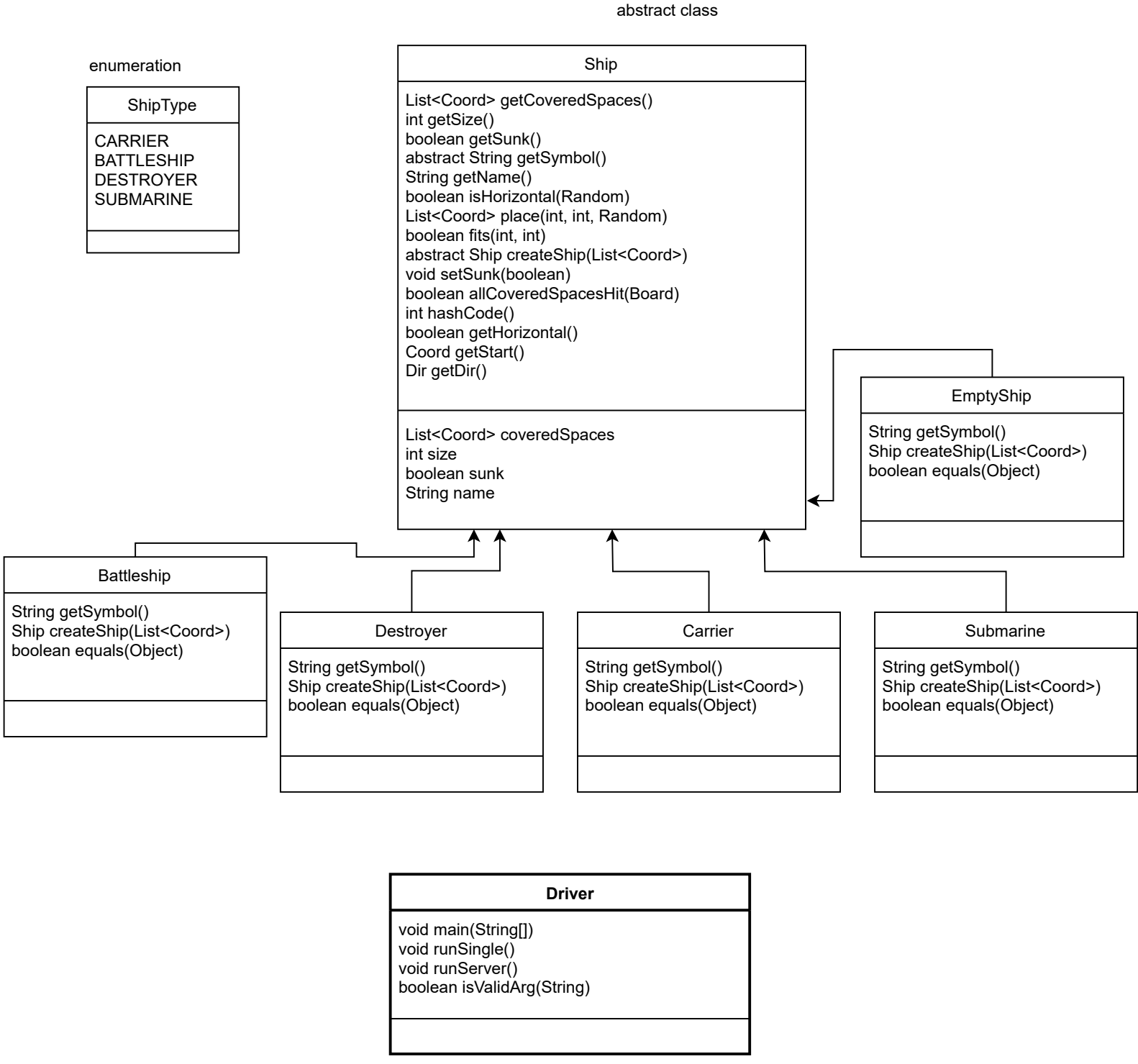
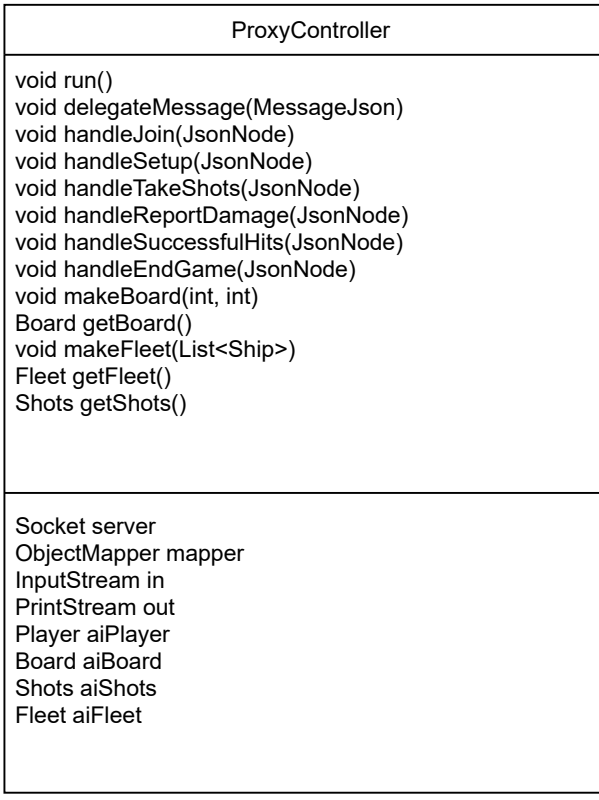
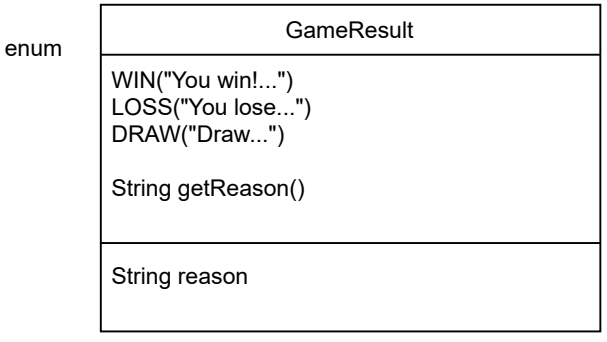
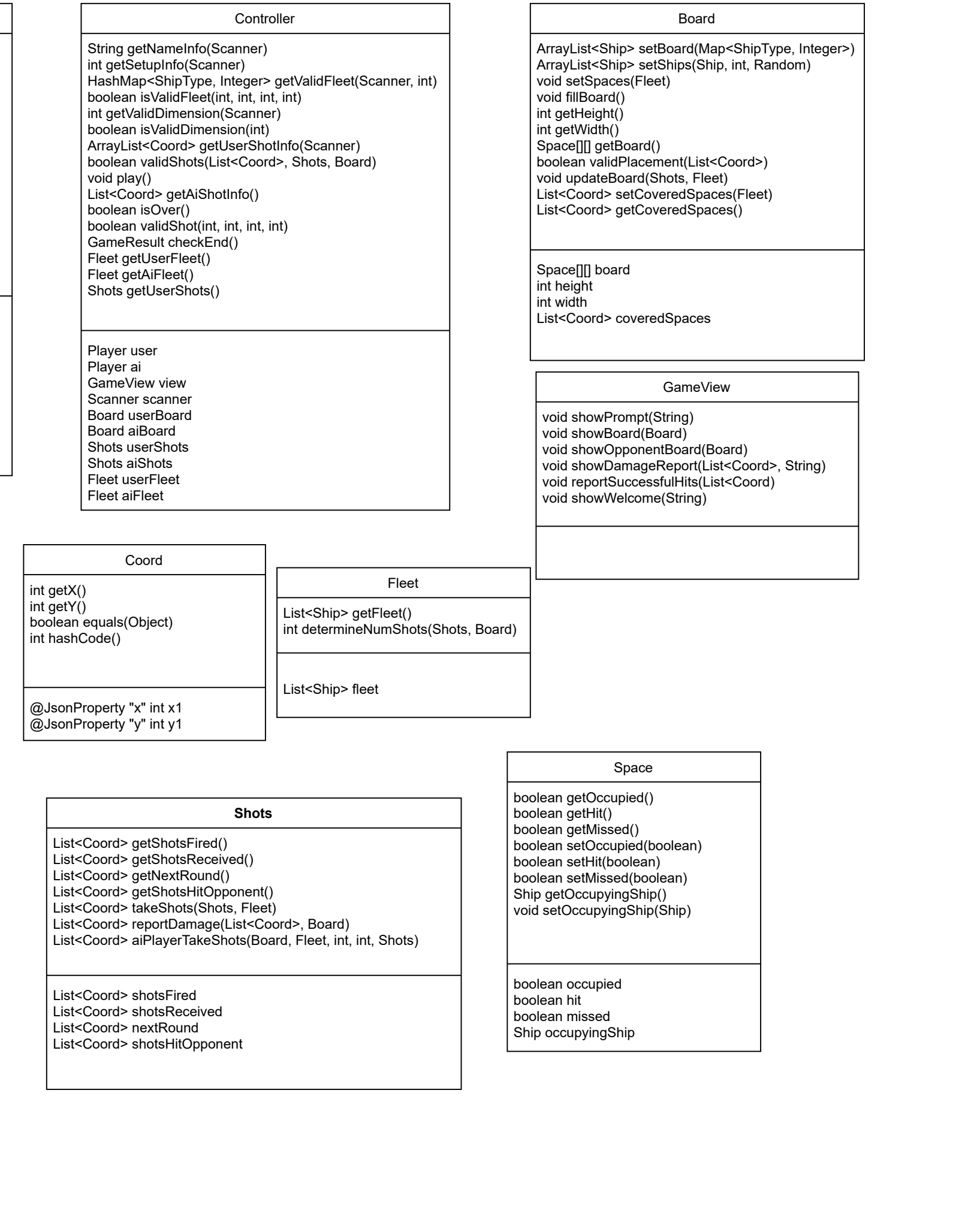
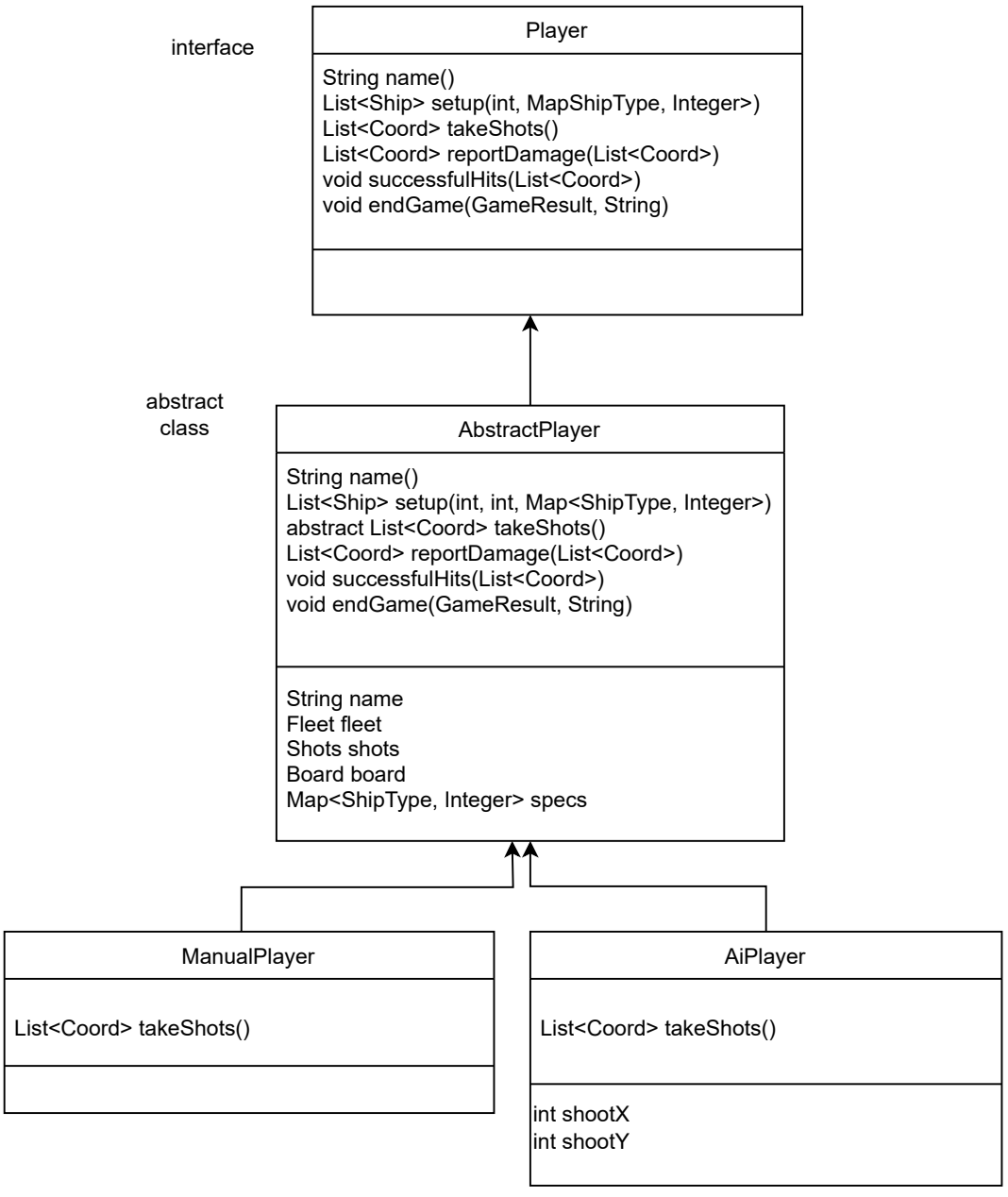

abstract class
enumeration
ShipType
CARRIER
BATTLESHIP
DESTROYER
SUBMARINE
Ship
List<Coord> getCoveredSpaces()
int getSize()
boolean getSunk()
abstract String getSymbol()
String getName()
boolean isHorizontal(Random)
List<Coord> place(int, int, Random)
boolean fits(int, int)
abstract Ship createShip(List<Coord>)
void setSunk(boolean)
boolean allCoveredSpacesHit(Board)
int hashCode()
boolean getHorizontal()
Coord getStart()
Dir getDir()
List<Coord> coveredSpaces
int size
boolean sunk
String name
EmptyShip
String getSymbol()
Ship createShip(List<Coord>)
boolean equals(Object)
Battleship
String getSymbol()
Ship createShip(List<Coord>)
boolean equals(Object)
Destroyer
String getSymbol()
Ship createShip(List<Coord>)
boolean equals(Object)
Carrier
String getSymbol()
Ship createShip(List<Coord>)
boolean equals(Object)
Submarine
String getSymbol()
Ship createShip(List<Coord>)
boolean equals(Object)
Driver
void main(String[])
void runSingle()
void runServer()
boolean isValidArg(String)


MessageJson
@JsonProperty ("method-name") String
@JsonProperty("arguments") JsonNode
ShipAdapter
JsonCreator
@JsonProperty ("start") Coord
@JsonProperty("length") int
@JsonProperty("direction") Direction
JoinJson
@JsonProperty ("name") String
@JsonProperty("game-mode") JsonNode
CoordinatesJson
@JsonProperty ("coordinates") List<Coord>
SetUpRecieiveJson
@JsonProperty ("height") int
@JsonProperty("width") int
@JsonProperty("fleet-spec") Map<ShipType, Integer>
public int carrier()
public int battleship()
public int destroyer()
public int submarine()
EndGameReceiveJson
@JsonProperty ("result") String
@JsonProperty ("reason") String
public GameResult getResult()
JsonUtils
public static JsonNode serializeRecord(Record )
SetUpSendJson
@JsonProperty("fleet-spec") Map<ShipType, Integer>
enum
Dir
HORIZONTAL
VERTICAL


ProxyController
void run()
void delegateMessage(MessageJson)
void handleJoin(JsonNode)
void handleSetup(JsonNode)
void handleTakeShots(JsonNode)
void handleReportDamage(JsonNode)
void handleSuccessfulHits(JsonNode)
void handleEndGame(JsonNode)
void makeBoard(int, int)
Board getBoard()
void makeFleet(List<Ship>)
Fleet getFleet()
Shots getShots()
Socket server
ObjectMapper mapper
InputStream in
PrintStream out
Player aiPlayer
Board aiBoard
Shots aiShots
Fleet aiFleet


enum
GameResult
WIN("You win!...")
LOSS("You lose...")
DRAW("Draw...")
String getReason()
String reason


Controller
String getNameInfo(Scanner)
int getSetupInfo(Scanner)
HashMap<ShipType, Integer> getValidFleet(Scanner, int)
boolean isValidFleet(int, int, int, int)
int getValidDimension(Scanner)
boolean isValidDimension(int)
ArrayList<Coord> getUserShotInfo(Scanner)
boolean validShots(List<Coord>, Shots, Board)
void play()
List<Coord> getAiShotInfo()
boolean isOver()
boolean validShot(int, int, int, int)
GameResult checkEnd()
Fleet getUserFleet()
Fleet getAiFleet()
Shots getUserShots()
Player user
Player ai
GameView view
Scanner scanner
Board userBoard
Board aiBoard
Shots userShots
Shots aiShots
Fleet userFleet
Fleet aiFleet
Board
ArrayList<Ship> setBoard(Map<ShipType, Integer>)
ArrayList<Ship> setShips(Ship, int, Random)
void setSpaces(Fleet)
void fillBoard()
int getHeight()
int getWidth()
Space[][] getBoard()
boolean validPlacement(List<Coord>)
void updateBoard(Shots, Fleet)
List<Coord> setCoveredSpaces(Fleet)
List<Coord> getCoveredSpaces()
Space[][] board
int height
int width
List<Coord> coveredSpaces
GameView
void showPrompt(String)
void showBoard(Board)
void showOpponentBoard(Board)
void showDamageReport(List<Coord>, String)
void reportSuccessfulHits(List<Coord)
void showWelcome(String)
Coord
int getX()
int getY()
boolean equals(Object)
int hashCode()
@JsonProperty "x" int x1
@JsonProperty "y" int y1
Fleet
List<Ship> getFleet()
int determineNumShots(Shots, Board)
List<Ship> fleet
Space
boolean getOccupied()
boolean getHit()
boolean getMissed()
boolean setOccupied(boolean)
boolean setHit(boolean)
boolean setMissed(boolean)
Ship getOccupyingShip()
void setOccupyingShip(Ship)
boolean occupied
boolean hit
boolean missed
Ship occupyingShip
Shots
List<Coord> getShotsFired()
List<Coord> getShotsReceived()
List<Coord> getNextRound()
List<Coord> getShotsHitOpponent()
List<Coord> takeShots(Shots, Fleet)
List<Coord> reportDamage(List<Coord>, Board)
List<Coord> aiPlayerTakeShots(Board, Fleet, int, int, Shots)
List<Coord> shotsFired
List<Coord> shotsReceived
List<Coord> nextRound
List<Coord> shotsHitOpponent


interface
Player
String name()
List<Ship> setup(int, MapShipType, Integer>)
List<Coord> takeShots()
List<Coord> reportDamage(List<Coord>)
void successfulHits(List<Coord>)
void endGame(GameResult, String)
abstract
class
AbstractPlayer
String name()
List<Ship> setup(int, int, Map<ShipType, Integer>)
abstract List<Coord> takeShots()
List<Coord> reportDamage(List<Coord>)
void successfulHits(List<Coord>)
void endGame(GameResult, String)
String name
Fleet fleet
Shots shots
Board board
Map<ShipType, Integer> specs
ManualPlayer
List<Coord> takeShots()
AiPlayer
List<Coord> takeShots()
int shootX
int shootY