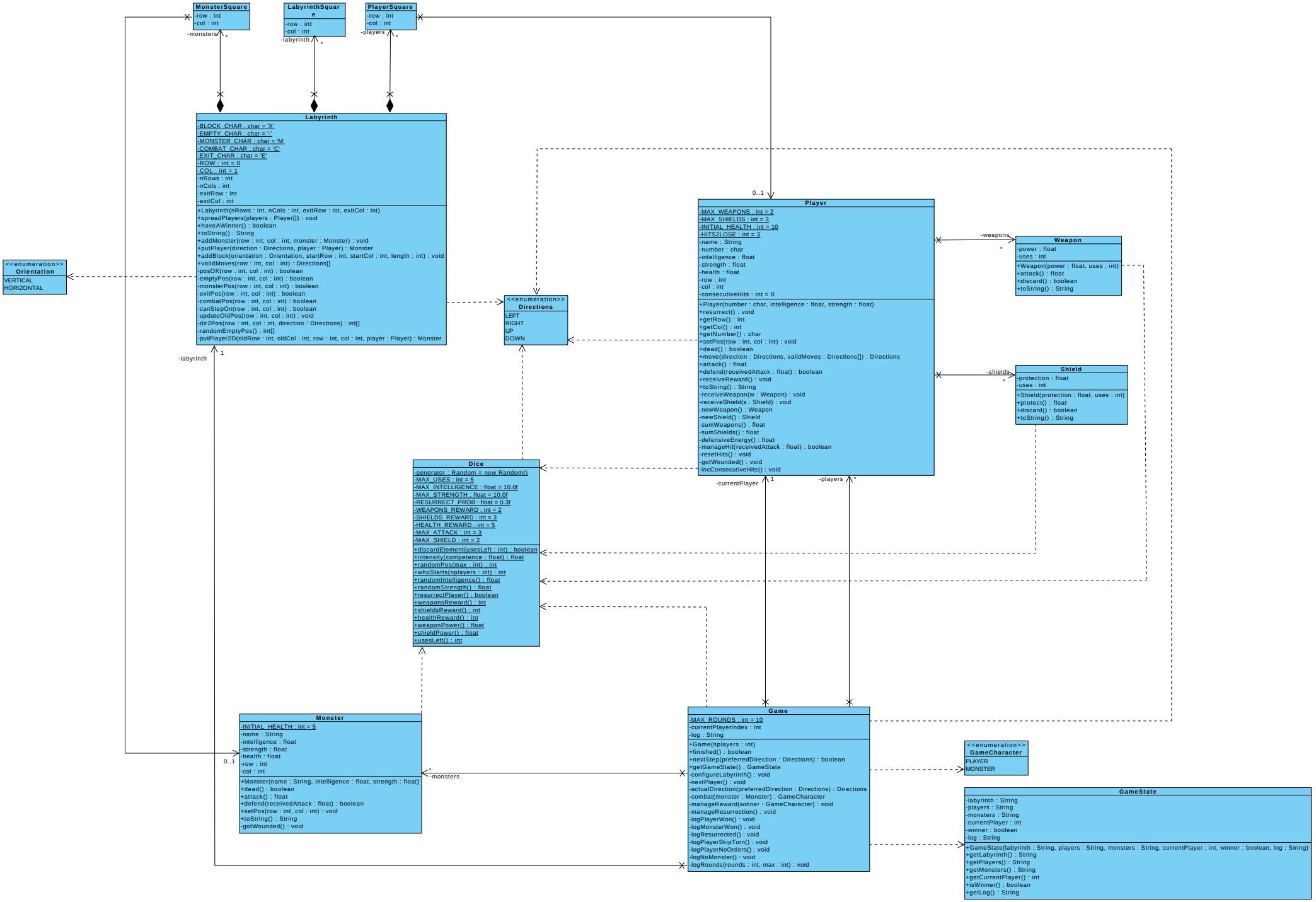
MonsterSquare
-row : int
-col : int
LabyrinthSquare
-row : int
-col : int
PlayerSquare
-row : int
-col : int
-monsters
-labyrinth
-players
Labyrinth
-BLOCK_CHAR : char = 'X'
-EMPTY_CHAR : char = '-'
-MONSTER_CHAR : char = 'M'
-COMBAT_CHAR : char = 'C'
-EXIT_CHAR : char = 'E'
-ROW : int = 0
-COL : int = 1
-nRows : int
-nCols : int
-exitRow : int
-exitCol : int
+Labyrinth(nRows : int, nCols : int, exitRow : int, exitCol : int)
+spreadPlayers(players : Player[]) : void
+haveAWinner() : boolean
+toString() : String
+addMonster(row : int, col : int, monster : Monster) : void
+putPlayer(direction : Directions, player : Player) : Monster
+addBlock(orientation : Orientation, startRow : int, startCol : int, length : int) : void
+validMoves(row : int, col : int) : Directions[]
-posOK(row : int, col : int) : boolean
-emptyPos(row : int, col : int) : boolean
-monsterPos(row : int, col : int) : boolean
-exitPos(row : int, col : int) : boolean
-combatPos(row : int, col : int) : boolean
-canStepOn(row : int, col : int) : boolean
-updateOldPos(row : int, col : int) : void
-dir2Pos(row : int, col : int, direction : Directions) : int[]
-randomEmptyPos() : int[]
-putPlayer2D(oldRow : int, oldCol : int, row : int, col : int, player : Player) : Monster
<<enumeration>>
Orientation
VERTICAL
HORIZONTAL
<<enumeration>>
Directions
LEFT
RIGHT
UP
DOWN
0..1
Player
-MAX_WEAPONS : int = 2
-MAX_SHIELDS : int = 3
-INITIAL_HEALTH : int = 10
-HITS2LOSE : int = 3
-name : String
-number : char
-intelligence : float
-strength : float
-health : float
-row : int
-col : int
-consecutiveHits : int = 0
+Player(number : char, intelligence : float, strength : float)
+resurrect() : void
+getRow() : int
+getCol() : int
+getNumber() : char
+setPos(row : int, col : int) : void
+dead() : boolean
+move(direction : Directions, validMoves : Directions[]) : Directions
+attack() : float
+defend(receivedAttack : float) : boolean
+receiveReward() : void
+toString() : String
-receiveWeapon(w : Weapon) : void
-receiveShield(s : Shield) : void
-newWeapon() : Weapon
-newShield() : Shield
-sumWeapons() : float
-sumShields() : float
-defensiveEnergy() : float
-manageHit(receivedAttack : float) : boolean
-resetHits() : void
-gotWounded() : void
-incConsecutiveHits() : void
-weapons
*
Weapon
-power : float
-uses : int
+Weapon(power : float, uses : int)
+attack() : float
+discard() : boolean
+toString() : String
-shields
*
Shield
-protection : float
-uses : int
+Shield(protection : float, uses : int)
+protect() : float
+discard() : boolean
+toString() : String
Dice
-generator : Random = new Random()
-MAX_USES : int = 5
-MAX_INTELLIGENCE : float = 10.0f
-MAX_STRENGTH : float = 10.0f
-RESURRECT_PROB : float = 0.3f
-WEAPONS_REWARD : int = 2
-SHIELDS_REWARD : int = 3
-HEALTH_REWARD : int = 5
-MAX_ATTACK : int = 3
-MAX_SHIELD : int = 2
+discardElement(usesLeft : int) : boolean
+intensity(competence : float) : float
+randomPos(max : int) : int
+whoStarts(nplayers : int) : int
+randomIntelligence() : float
+randomStrength() : float
+resurrectPlayer() : boolean
+weaponsReward() : int
+shieldsReward() : int
+healthReward() : int
+weaponPower() : float
+shieldPower() : float
+usesLeft() : int
-currentPlayer
1
-players
*
Monster
-INITIAL_HEALTH : int = 5
-name : String
-intelligence : float
-strength : float
-health : float
-row : int
-col : int
+Monster(name : String, intelligence : float, strength : float)
+dead() : boolean
+attack() : float
+defend(receivedAttack : float) : boolean
+setPos(row : int, col : int) : void
+toString() : String
-gotWounded() : void
0..1
*
-monsters
Game
-MAX_ROUNDS : int = 10
-currentPlayerIndex : int
-log : String
+Game(nplayers : int)
+finished() : boolean
+nextStep(preferredDirection : Directions) : boolean
+getGameState() : GameState
-configureLabyrinth() : void
-nextPlayer() : void
-actualDirection(preferredDirection : Directions) : Directions
-combat(monster : Monster) : GameCharacter
-manageReward(winner : GameCharacter) : void
-manageResurrection() : void
-logPlayerWon() : void
-logMonsterWon() : void
-logResurrected() : void
-logPlayerSkipTurn() : void
-logPlayerNoOrders() : void
-logNoMonster() : void
-logRounds(rounds : int, max : int) : void
<<enumeration>>
GameCharacter
PLAYER
MONSTER
GameState
-labyrinth : String
-players : String
-monsters : String
-currentPlayer : int
-winner : boolean
-log : String
+GameState(labyrinth : String, players : String, monsters : String, currentPlayer : int, winner : boolean, log : String)
+getLabyrinth() : String
+getPlayers() : String
+getMonsters() : String
+getCurrentPlayer() : int
+isWinner() : boolean
+getLog() : String
-labyrinth
1