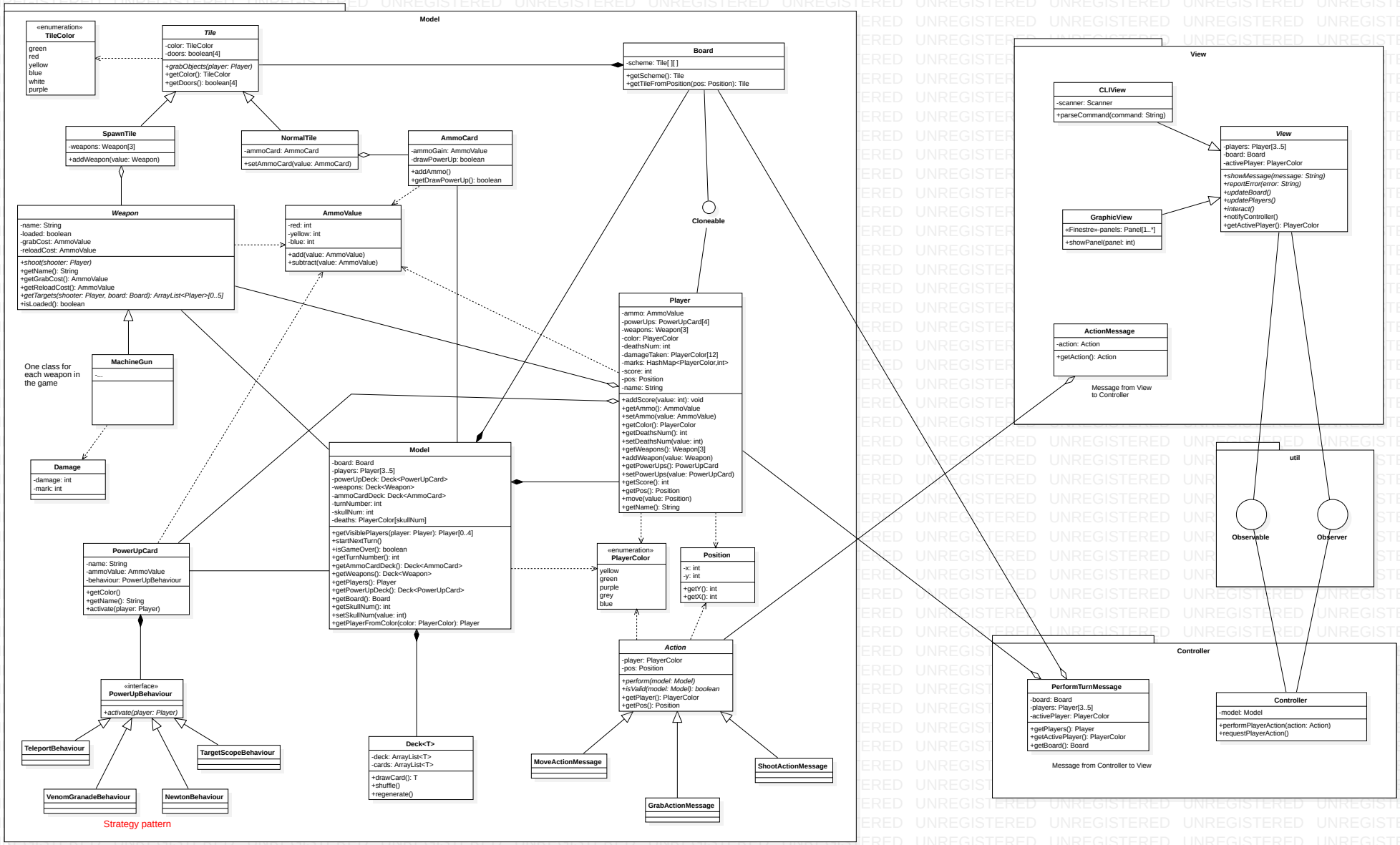
Model.Main


Model
«enumeration»
TileColor
green
red
yellow
blue
white
purple
Tile
-color: TileColor
-doors: boolean[4]
+grabObjects(player: Player)
+getColor(): TileColor
+getDoors(): boolean[4]
Board
-scheme: Tile[ ][ ]
+getScheme(): Tile
+getTileFromPosition(pos: Position): Tile
SpawnTile
-weapons: Weapon[3]
+addWeapon(value: Weapon)
NormalTile
-ammoCard: AmmoCard
+setAmmoCard(value: AmmoCard)
AmmoCard
-ammoGain: AmmoValue
-drawPowerUp: boolean
+addAmmo()
+getDrawPowerUp(): boolean
Weapon
-name: String
-loaded: boolean
-grabCost: AmmoValue
-reloadCost: AmmoValue
+shoot(shooter: Player)
+getName(): String
+getGrabCost(): AmmoValue
+getReloadCost(): AmmoValue
+getTargets(shooter: Player, board: Board): ArrayList<Player>[0..5]
+isLoaded(): boolean
AmmoValue
-red: int
-yellow: int
-blue: int
+add(value: AmmoValue)
+subtract(value: AmmoValue)
Cloneable
Player
-ammo: AmmoValue
-powerUps: PowerUpCard[4]
-weapons: Weapon[3]
-color: PlayerColor
-deathsNum: int
-damageTaken: PlayerColor[12]
-marks: HashMap<PlayerColor,int>
-score: int
-pos: Position
-name: String
+addScore(value: int): void
+getAmmo(): AmmoValue
+setAmmo(value: AmmoValue)
+getColor(): PlayerColor
+getDeathsNum(): int
+setDeathsNum(value: int)
+getWeapons(): Weapon[3]
+addWeapon(value: Weapon)
+getPowerUps(): PowerUpCard
+setPowerUps(value: PowerUpCard)
+getScore(): int
+getPos(): Position
+move(value: Position)
+getName(): String
One class for each weapon in the game
MachineGun
-...
Damage
-damage: int
-mark: int
Model
-board: Board
-players: Player[3..5]
-powerUpDeck: Deck<PowerUpCard>
-weapons: Deck<Weapon>
-ammoCardDeck: Deck<AmmoCard>
-turnNumber: int
-skullNum: int
-deaths: PlayerColor[skullNum]
+getVisiblePlayers(player: Player): Player[0..4]
+startNextTurn()
+isGameOver(): boolean
+getTurnNumber(): int
+getAmmoCardDeck(): Deck<AmmoCard>
+getWeapons(): Deck<Weapon>
+getPlayers(): Player
+getPowerUpDeck(): Deck<PowerUpCard>
+getBoard(): Board
+getSkullNum(): int
+setSkullNum(value: int)
+getPlayerFromColor(color: PlayerColor): Player
«enumeration»
PlayerColor
yellow
green
purple
grey
blue
Position
-x: int
-y: int
+getY(): int
+getX(): int
PowerUpCard
-name: String
-ammoValue: AmmoValue
-behaviour: PowerUpBehaviour
+getColor()
+getName(): String
+activate(player: Player)
Action
-player: PlayerColor
-pos: Position
+perform(model: Model)
+isValid(model: Model): boolean
+getPlayer(): PlayerColor
+getPos(): Position
«interface»
PowerUpBehaviour
+activate(player: Player)
TeleportBehaviour
TargetScopeBehaviour
VenomGranadeBehaviour
NewtonBehaviour
Strategy pattern
Deck<T>
-deck: ArrayList<T>
-cards: ArrayList<T>
+drawCard(): T
+shuffle()
+regenerate()
MoveActionMessage
GrabActionMessage
ShootActionMessage
View
CLIView
-scanner: Scanner
+parseCommand(command: String)
View
-players: Player[3..5]
-board: Board
-activePlayer: PlayerColor
+showMessage(message: String)
+reportError(error: String)
+updateBoard()
+updatePlayers()
+interact()
+notifyController()
+getActivePlayer(): PlayerColor
GraphicView
«Finestre»-panels: Panel[1..*]
+showPanel(panel: int)
ActionMessage
-action: Action
+getAction(): Action
Message from View to Controller
util
Observable
Observer
Controller
PerformTurnMessage
-board: Board
-players: Player[3..5]
-activePlayer: PlayerColor
+getPlayers(): Player
+getActivePlayer(): PlayerColor
+getBoard(): Board
Message from Controller to View
Controller
-model: Model
+performPlayerAction(action: Action)
+requestPlayerAction()