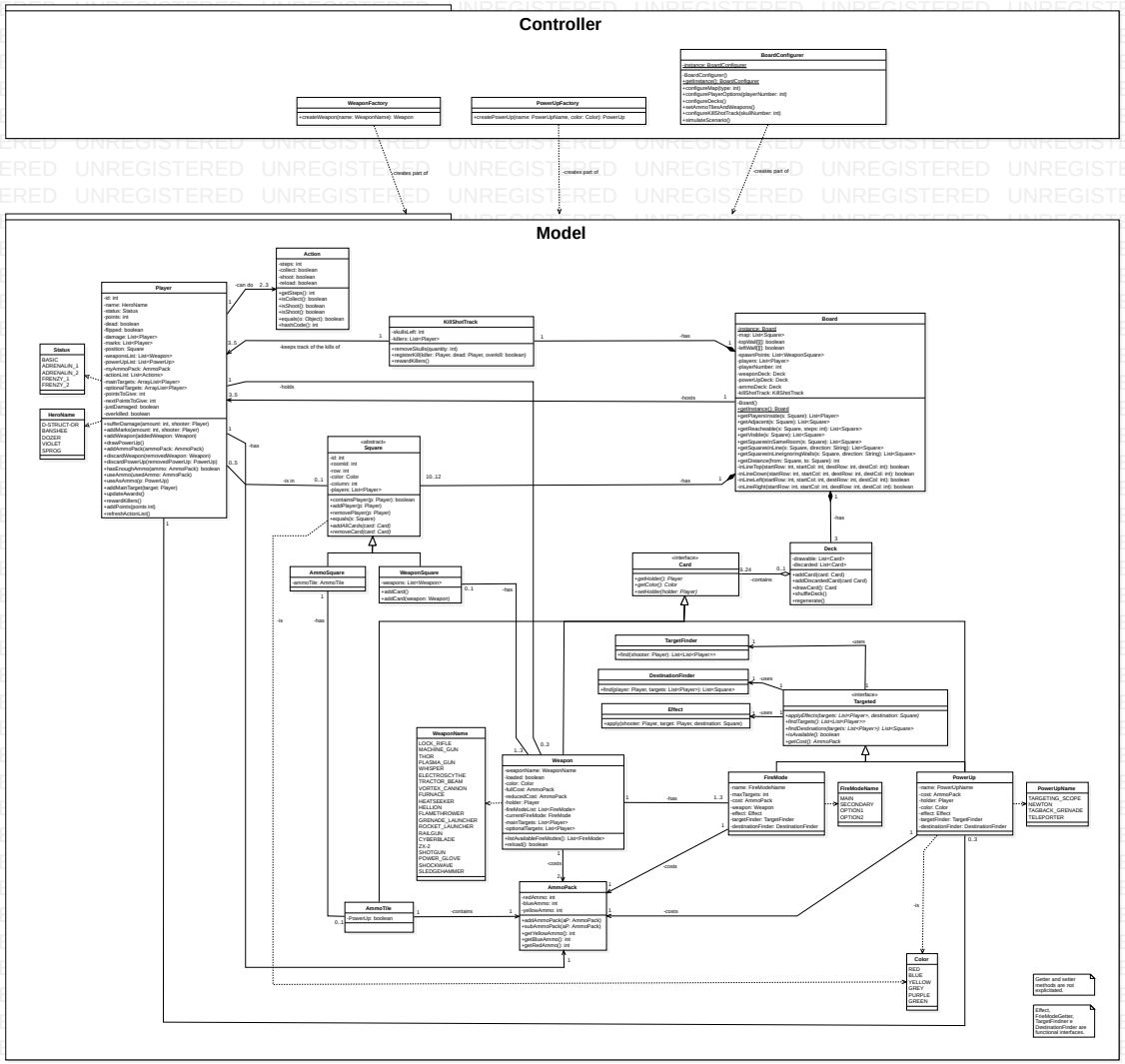
Controller
BoardConfigurer
-instance: BoardConfigurer
-BoardConfigurer()
+getInstance(): BoardConfigurer
+configureMap(type: int)
+configurePlayerOptions(playerNumber: int)
+configureDecks()
+setAmmoTilesAndWeapons()
+configure4SDoubleTrackskillNumber: int)
+simulateScenario()
WeaponFactory
+createWeapon(name: WeaponName): Weapon
PowerUpFactory
+createPowerUp(name: PowerUpName, color: Color): PowerUp
-creates part of
-creates part of
-creates part of
Model
Action
-steps: int
-collect: boolean
-shoot: boolean
-reload: boolean
+getSteps(): int
+isCollect(): boolean
+isShoot(): boolean
+isShoot(): boolean
+equals(o: Object): boolean
+hashCode(): int
Player
-id: int
-name: HeroName
-status: Status
-points: int
-dead: boolean
-flipped: boolean
-damage: List<Player>
-marks: List<Player>
-position: Square
-weapons:List: List<Weapon>
-myPowerUps: List<PowerUp>
-myAmmoPack: AmmoPack
-actionList: List<Action>
-mainTargets: ArrayList<Player>
-optionalTargets: ArrayList<Player>
-pointsToGive: int
-newPointsToGive: int
-justDamaged: boolean
-overkilled: boolean
+setNeDamage(amount: int, shooter: Player)
+addMarks(amount: int, shooter: Player)
+addWeapon(addedWeapon: Weapon)
+drawPowerUp()
+addAmmoPack(ammoPack: AmmoPack)
+discardWeapon(removedWeapon: Weapon)
+discardPowerUp(removedPowerUp: PowerUp)
+hasEnoughAmmo(ammo: AmmoPack): boolean
+useAmmo(ammo: AmmoPack)
+useAnAmmo(p: PowerUp)
+addMainTarget(target: Player)
+updateAwards()
+rewardKillers()
+addPoints(points int)
+refreshActionList()
-can do
2..3
1
Status
BASIC
ADRENALIN_1
ADRENALIN_2
FRENZY_1
FRENZY_2
HeroName
D-STRUCT-OR
BANSHEE
DOZER
VIOLET
SPROG
0..3
-keeps track of the kills of
1
KillShotTrack
-skullsLeft: int
-killers: List<Player>
+removeSkulls(quantity: int)
+registerKill(killer: Player, dead: Player, overkill: boolean)
+rewardKillers()
1
-has
Board
-instance: Board
-map: List<Square>
-isBWMS: boolean
-isFrenzy: boolean
-spawnFinder: ListWeaponSquare>
-players: List<Player>
-playerNumber: int
-weaponDeck: Deck
-powerUpDeck: Deck
-ammoDeck: Deck
-killShotTrack: KillShotTrack
-Board()
+getInstance(): Board
+getPlayersInside(s: Square): List<Player>
+getAdjacent(s: Square): List<Square>
+getReachable(s: Square, step: int): List<Square>
+getVisible(s: Square): List<Square>
+getSquaresInGameRoom(s: Square): List<Square>
+getSquaresInLine(s: Square, direction: String): List<Square>
+getSquaresInLongsnorkVisible(s: Square, direction: String): List<Square>
+isLineOfSight(sourceRow: int, sourceCol: int, destRow: int, destCol: int): boolean
+isLineDown(startRow: int, startCol: int, destRow: int, destCol: int): boolean
+isLineLeft(startRow: int, startCol: int, destRow: int, destCol: int): boolean
+isLineRight(startRow: int, startCol: int, destRow: int, destCol: int): boolean
-has
1
-has
1
-hosts
1
-has
1
-has
1
-is in
0..1
10..12
«abstract»
Square
-id: int
-roomId: int
-row: int
-color: Color
-column: int
-players: List<Player>
+containsPlayer(p: Player): boolean
+addPlayer(p: Player)
+removePlayer(p: Player)
+equals(s: Square)
+addMARCards(card: Card)
+removeCard(card: Card)
AmmoSquare
-ammoTile: AmmoTile
WeaponSquare
-weapons: List<Weapon>
+addCard()
+addCard(weapon: Weapon)
0..1
-has
«interface»
Card
+getHolder(): Player
+getColor(): Color
+setHolder(holder: Player)
1..24
0..1
-contains
Deck
-drawable: List<Card>
-discarded: List<Card>
+addCard(card: Card)
+addDiscardedCard(card Card)
+drawCard(): Card
+shuffleDeck()
+regenerate()
TargetFinder
+find(shooter: Player): List<List<Player>>
-uses
DestinationFinder
+find(player: Player, targets: List<Player>): List<Square>
-uses
Effect
+apply(shooter: Player, target: Player, destination: Square)
-uses
«interface»
Targeted
+applyEffects(targets: List<Player>, destination: Square)
+findTargets(): List<List<Player>>
+findDestinations(targets: List<Player>): List<Square>
+isAvailable(): boolean
+getCost(): AmmoPack
WeaponName
LOCK_RIFLE
MACHINE_GUN
THOR
PLASMA_GUN
WHISPER
ELECTROSCYTHE
TRACTOR_BEAM
VORTEX_CANNON
FURNACE
HEATSEEKER
HELLION
FLAMETHROWER
GRENADE_LAUNCHER
ROCKET_LAUNCHER
RAILGUN
CYBERBLADE
ZX-2
SHOTGUN
POWER_GLOVE
SHOCKWAVE
SLEDGEHAMMER
Weapon
-weaponName: WeaponName
-loaded: boolean
-color: Color
-fullCost: AmmoPack
-reloadCost: AmmoPack
-holder: Player
-fireModeList: List<FireMode>
-currentFireMode: FireMode
-mainTargets: List<Player>
-optionalTargets: List<Player>
+listAvailableFireModes(): List<FireMode>
+reload(): boolean
1..3
0..3
-has
1
1..3
-costs
1
-costs
1
FireMode
-name: FireModeName
-maxTargets: int
-cost: AmmoPack
-weapon: Weapon
-effect: Effect
-targetFinder: TargetFinder
-destinationFinder: DestinationFinder
FireModeName
MAIN
SECONDARY
OPTION1
OPTION2
PowerUp
-name: PowerUpName
-cost: AmmoPack
-holder: Player
-color: Color
-effect: Effect
-targetFinder: TargetFinder
-destinationFinder: DestinationFinder
PowerUpName
TARGETING_SCOPE
NEWTON
TAGBACK_GRENADE
TELEPORTER
0..3
-is
AmmoTile
-PowerUp: boolean
1
-contains
1
AmmoPack
-redAmmo: int
-blueAmmo: int
-yellowAmmo: int
+addAmmoPack(aP: AmmoPack)
+subAmmoPack(aP: AmmoPack)
+getYellowAmmo(): int
+getBlueAmmo(): int
+getRedAmmo(): int
1
-costs
1
Color
RED
BLUE
YELLOW
GREY
PURPLE
GREEN
Getter and setter methods are not implemented.
Effect, FireModeGetter, TargetFinder e DestinationFinder are functional interfaces.
-holds