DungeonController
- squares: GridPane
- initialEntities: List<ImageView>
- inventory: GridPane
- treasureCounter: TextField
- inventoryField: TextField
- goalField: TextField
- mainStage: Stage
- player: Player
- dungeon: Dungeon

+ initialize(): void
+ handleKeyPress(KeyEvent): void
+ removeItem(): void
+ updateStatus(): void
+ loadDungeon(): void
+ dropItem(Entity): void
+ bombState(): void
+ invincibleState(Invincibility): void
+ gameOver(): void
+ setMainStage(): void

Player
- dungeon: Dungeon
- carryOns: Entity
- treasure: int
- invincibility: Invincibility
- dead: boolean
- life: int

+ getInvincibility(): Invincibility
+ dropEntity(): void
+ killEnemy(): int
+ isDead(): boolean
+ moveUp(): int
+ moveDown(): int
+ moveRight(): int
+ moveLeft(): int
+ getCarryOns(): Entity
+ getDurability(): int
+ getTreasure(): int
+ setInvincibility(Invincibility): void
+ getDungeon(): Dungeon
+ setBomb(): void
+ setDead(boolean): void
+ setCarryOns(Entity): void
+ setTreasure(Entity): void

Dungeon
- width: int
- height: int
- entities: List<Entity>
- removedEntity: List<Entity>
- player: Player
- goals: JSONObject
- dungeonGoal: Goals

+ getWidth(): int
+ getHeight(): int
+ getPlayer(): Player
+ setPlayer(Player): void
+ addEntity(Entity): void
+ getEntities(): List<Entity>
+ setEntities(List<Entity>): void
+ checkGoal(Goals): boolean
+ setGoals(JSONArray): void
+ removeEntity(Entity): void
+ checkEnemyGoal(): boolean
+ checkTreasureGoal(): boolean
+ explode(Bomb): void
+ getEntity(int, int): Entity
+ getGoals(): String
+ getRemovedEntity(): List<Entity>
+ addRemovedEntity(Entity): void
+ bombState(): void
+ getEntity(int,int): Entity
+ setDungeonGoal(Goals): void
+ getDungeonGoal(): Goals
+ removeRemoved(): void
+ goalToJSON(JSONObject): void
+ checkDead(): boolean
+ checkOrGoalComposite(CompositeGoal): boolean
+ checkAndGoalComposite(CompositeGoal): boolean
+ checkExitGoal(): boolean
+ checkBoulderGoal(): boolean

DungeonLoader
- json: JSONObject
- keyId: int
- doorId: int

+ load(): Dungeon
+ loadEntity(Dungeon, JSONObject): void
+ onLoad(Invincibility): void
+ onLoad(Sword): void
+ onLoad(Enemy): void
+ onLoad(Bomb): void
+ onLoad(Switch): void
+ onLoad(Boulder): void
+ onLoad(Key): void
+ onLoad(Door): void
+ onLoad(Treasure): void
+ onLoad(Exit): void
+ onLoad(Wall): void
+ onLoad(Entity): void
+ onLoad(Hound): void

DungeonControllerLoader
- entities: List<ImageView>
- playerImage: Image
- wallImage: exitImage
- exitImage: Image
- treasureImage: Image
- doorImage: Image
- keyImage: Image
- boulderImage: Image
- switchImage: Image
- unlitBombImage: Image
- enemyImage: Image
- houndImage: Image
- swordImage: Image
- invincibilityImage: Image

+ addEntity(Entity, ImageVie
+ trackPosition(Entity, Node)
+ loadController(): DungeonI

Entity
- x: IntegerProperty
- x: IntegerProperty
- image: ImageView

+ x(): IntegerProperty
+ y(): IntegerProperty
+ getY(): int
+ getX(): int
+ getImage(): ImageView
+ setImage(ImageView): voi
+ interact(Player): Entity
+ drop(): Entity
+ pass(): boolean
+ remove(): boolean

Wall
+ pass(): boolean

Exit
+ interact(Player): Entity

Treasure
+ interact(Player): Entity
+ remove(): boolean
+ run(): void

Sword
- durability: int
+ reduceDurability(): void
+ interact(Player): Entity
+ getDurability(): int
+ setDurability(int): void
+ drop(): Entity
+ remove(): boolean
+ toString(): String

Boulder
+ checkBoulder(int, int, List<Entity>): boolean
+ interact(Player): Entity
+ moveBoulder(int, int): void

LockedDoor
- id: int
- isOpen: boolean
+ getId(): int
+ setId(int): void
+ isOpen(): boolean
+ setOpen(boolean): void
+ interact(Player): Entity
+ openDoor(): void

Switch
- switched: boolean
+ interact(Player): Entity
+ boulderSwitch(): void
+ isSwitched(): boolean
+ pass(): boolean

Key
- id: int
+ getId(): int
+ interact(Player): Entity
+ setId(int): void
+ remove(): boolean
+ toString(): String
+ drop(): Entity

Bomb
- unlitBomb: BombState
- litBomb1: BombState
- litBomb2: BombState
- litBomb3: BombState
- explodingBomb: BombState
- state: BombState
- countdown: int

+ getCountdown(): int
+ setCountdown(int): void
+ countDownBomb(): void
+ getUnlitBomb(): BombState
+ getLitBomb1(): BombState
+ getLitBomb2(): BombState
+ getLitBomb3(): BombState
+ getExplodingBomb(): BombState
+ toString(): String
+ interact(Player): Entity
+ remove(): boolean
+ setExplodingBomb(BombState): void
+ getState(): BombState
+ setState(BombState): void

Invincibility
- countdown: int
- temp: Player

+ countdownInvincibility(): void
+ getCountdown(): int
+ setCountdown(int): void
+ interact(Player): Entity
+ remove(): boolean
+ toString(): String

Enemy
+ interact(Player): Entity
+ remove(): boolean

Hound
+ interact(Player): Entity
+ remove(): boolean

BotAutoMove
- Dungeon: dungeon
- player: Player
- enemy: Entity
- timeDelay: int

+ run(): void
+ getAllEntitiesCoordinates(): ArrayList<Point:

LeafGoal
- goal: String

**«Interface»
Goals**
+ getGoal(): void

CompositeGoal
- goalList: List<Goals>
- logicalOperator: String
+ goalGenerate(JSONArray): void

**«interface»
AutoMoveAction**
+ autoMove(ArrayList<Point>,Point,Point,int,int): Point

EnemyChase
+ autoMove(ArrayList<Point:
+ BFS(Point,Point,ArrayList:

EnemyRunAway
+ autoMove(ArrayList<Point:
+ BFS(Point,Point,ArrayList:

**«interface»
BombState**
+ explode(): void
+ countdown(): void

UnlitBomb
+ bomb: Bomb

LitBomb1
+ bomb: Bomb

LitBomb2
+ bomb: Bomb

LitBomb3
+ bomb: Bomb

PostExplodingBomb
+ bomb: Bomb

ExplodingBomb
+ bomb: Bomb