GameController

- backBtn: Button
- boardNameLabel: Label
- gameGrid: GridPane

+ initialize(): void
+ handleBackBtn(): void
+ updateBoard(): void
+ addEntityImages(StackPane, St

<<interface>> CanMoveOnto

+ canMoveOnto(Board, BoardEntity): boolean

1..*   1..*

CMODecorator

- cmo: CanMoveOnto

1

<<enumeration>> EntityType

- value: int

+ valueOf(int): EntityType

+ getValue(): int

1

1

DungeonFactory

+ makeBoardEntity(int,int,int): BoardEntity

1

1

DungeonCreator

LevelS

- boards: ArrayLis
- startIndex: int
- boardType: Strir
- levelsPane: Pan
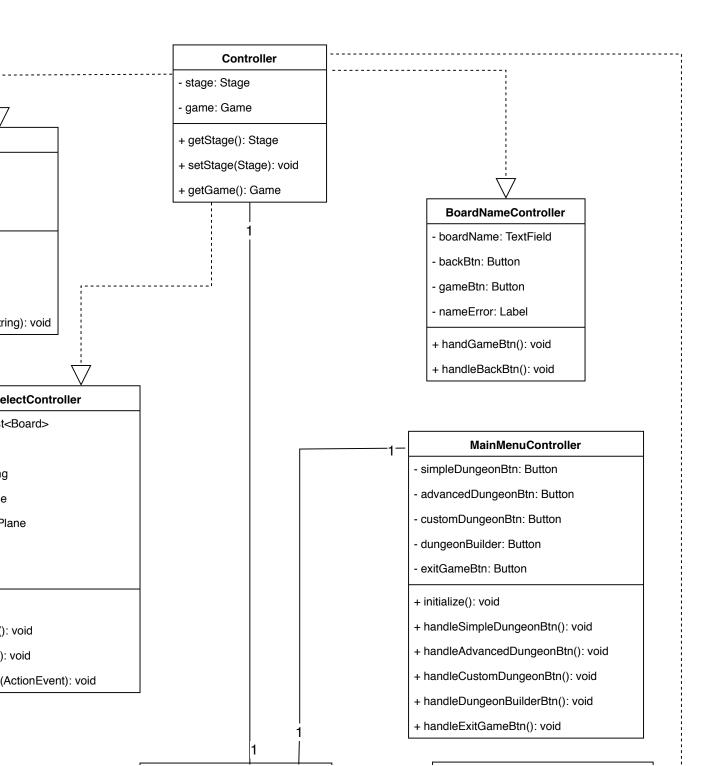- levelsGrid: GridF
- backBtn: Button
- nextBtn: Button

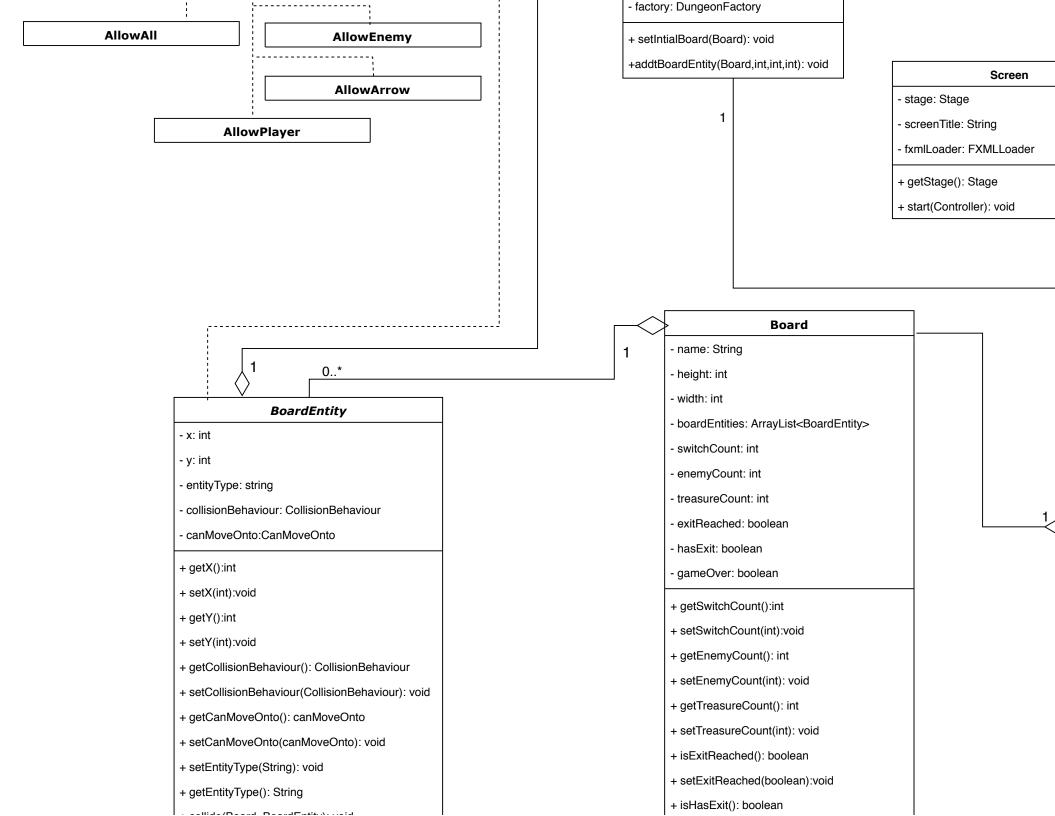+ initialize(): void
+ handleBackBtn(
+ handleNextBtn(
+ handleLevelBtn

AllowNone

AllowBoulder

Controller

- stage: Stage
- game: Game

+ getStage(): Stage
+ setStage(Stage): void
+ getGame(): Game

BoardNameController

- boardName: TextField
- backBtn: Button
- gameBtn: Button
- nameError: Label

+ handGameBtn(): void
+ handleBackBtn(): void

MainMenuController

- simpleDungeonBtn: Button
- advancedDungeonBtn: Button
- customDungeonBtn: Button
- dungeonBuilder: Button
- exitGameBtn: Button

+ initialize(): void
+ handleSimpleDungeonBtn(): void
+ handleAdvancedDungeonBtn(): void
+ handleCustomDungeonBtn(): void
+ handleDungeonBuilderBtn(): void
+ handleExitGameBtn(): void

electController

t<Board>

g
e
Plane

): void
): void
(ActionEvent): void

ring): void

1

1

1

1

AllowAll

AllowEnemy

AllowArrow

AllowPlayer

- factory: DungeonFactory

+ setIntialBoard(Board): void

+addtBoardEntity(Board,int,int,int): void

Screen

- stage: Stage

- screenTitle: String

- fxmlLoader: FXMLLoader

+ getStage(): Stage

+ start(Controller): void

BoardEntity

- x: int

- y: int

- entityType: string

- collisionBehaviour: CollisionBehaviour

- canMoveOnto:CanMoveOnto

+ getX():int

+ setX(int):void

+ getY():int

+ setY(int):void

+ getCollisionBehaviour(): CollisionBehaviour

+ setCollisionBehaviour(CollisionBehaviour): void

+ getCanMoveOnto(): canMoveOnto

+ setCanMoveOnto(canMoveOnto): void

+ setEntityType(String): void

+ getEntityType(): String

Board

- name: String

- height: int

- width: int

- boardEntities: ArrayList<BoardEntity>

- switchCount: int

- enemyCount: int

- treasureCount: int

- exitReached: boolean

- hasExit: boolean

- gameOver: boolean

+ getSwitchCount():int

+ setSwitchCount(int):void

+ getEnemyCount(): int

+ setEnemyCount(int): void

+ getTreasureCount(): int

+ setTreasureCount(int): void

+ isExitReached(): boolean

+ setExitReached(boolean):void

+ isHasExit(): boolean

1

1

1

0..*

1

ProjectRunner

+ main(String[]): void

ProjectController

- game: Game

+ getGame(): Game

+ start(): void

+ makeNewBoard(String, int , int): void

+ addRequestedEntity(int,int, int): void

+ removeRequestedEntity(int,int): void

+ keyPressed(KeyEvent): void

+ keyReleased(KeyEvent): void

+ keyTyped(KeyEvent): void

1

1

1

1

1

1

1

Game

- advancedDungeons: ArrayList<Board>

- customDungeons: ArrayList<Board>

- simpleDungeons: ArrayList<Board>

- currentBoard: Board

- boardLoader: BoardLoader

- boardWriter: BoardWriter

- dungeonCreator : DungeonCreator

- gamePlayer: GamePlayer

+ getSimpleDungeons(): ArrayList<Board>

+ setSimpleDungeons(ArrayList<Board>): void

+ getAdvancedDungeons(): ArrayList<Board>

+ setAdvancedDungeons(ArrayList<Board>): void

+ getCustomDungeons(): ArrayList<Board>

+ setCustomDungeons(ArrayList<Board>): void

+getCurrentBoard(): Board

DungeonBuilderController

- draggedSprite: String

- boardName: String

- boardGridPane: GridPane

- backBtn: Button

- saveBtn: Button

- boardNameLabel: Label

- saveMessage: Label

- binIcon: ImageView

- playerSprite: ImageView

- wallSpriteL ImageView

- pitSprite: ImageView

- exitSprite: ImageView

- boulderSprite: ImageView

- switchSprite: ImageView

- doorSprite: ImageView

- keySprite: ImageView

Class Diagram (partial)

Top-left class (partially visible):
- + collide(Board, BoardEntity): void
- + update(Board): void
- + isEnemy(): boolean
- + isPlayer(): boolean

Top-right class (partially visible):
- + setHasExit(boolean):void
- + isGameOver(): boolean
- + setGameOver(boolean): void
- + getName(String): int
- + getHeight():int
- + getWidth():int
- + getBoardEntities(): ArrayList<BoardEntity>
- + getPlayer(): Player
- + getEntitiesAt(int,int)
- + getEntityTpesAt(int,int): ArrayList<String>
- + addBoardEntity(BoardEntity): boolean
- + removeBoardEntity(BoardEntity): void
- + validX(int): boolean
- + validY(int): boolean   1
- + canMoveOnto(BoardEntity,int,int): boolean
- + updateBoard(): void

<<interface>> CollisionBehaviour
- + collide(Board,BoardEntity, BoardEntity): void

Classes (left column):
- PickUpInvinciblityPotion
- MoveAroundCollisionBehaviour
- CollideWithTrap
- CollideWithPlayer
- CollideWithEnemy
- CollideWithFloorSwitch
- PitKillCollisionBehaviour
- CollideWithBoulderSwitch
- WinLevelBehaviour
- Pit

Classes (middle column):
- PickUpArrows
- PickUpKey
- PickUpTreasure
- PickUpUnlitBomb
- NoCollision
- PickUpHoverPotion
- PickUpSword

MovingEntity *(abstract)*
- - direction: Direction
- - movementBehaviour: MovementBehaviour

- + getDirection(): Direction
- + setDirection(Direction):
- + getMovementBehaviours():int
- + setMovementBehaviour(MovementBehaviour): void
- + moveUp(Board): boolean
- + moveDown(Board): boolean
- + moveLeft(Board): boolean
- + moveRight(Board): boolean
- + moveTo(Board,int,int): boolean

Right class (partially visible):
- - bom
- - arro
- - swo
- - hove
- - invir
- - keys

- + getl
- + getA
- + getS
- + getl
- + getl
- + drop

1   1

+setCurrentBoard(Board): void

+ setGamePlayer(GamePlayer): void

+getGamePlayer(): GamePlayer

+boardsEmpty(): boolean

+createNewBoard(String,int,int): void

+checkBoardName(String): boolean

+ addEntityToBoard(int,int,int): void

+ removeEntitiesAt(int,int): void

+ getCustomDungeonsByName(): Board

+ saveCustomBoard(int): void

+ loadAllBoards(): void

+ loadBoards(String): void

- getLevel(int): boolean

+ startGame(int): void

+ shootArrow(): boolean

+ dorpBomb(): void

+ movePlayer(String):boolean

+ newTurn(): void

- treasureSprite: ImageView

- bombSprite: ImageView

- swordSprite: ImageView

- hunterSprite: ImageView

- arrowSprite: ImageView

- houndSprite: ImageView

- strategistSprite: ImageView

-cowardSprite: ImageView

- ipSprite: ImageView

- hpSprite: ImageView

+ getDraggedSprite(String): void

+ setDraggedSprite(String): void

+dragDetected(MouseEvent, ImageView): void

+dragPlacedItem(MouseEvent, ImageView): vo

+ addEntity(ImageView, StackPane): void

+ initialise(): void

+ handleSaveBtn(): void

+ handleBackBtn(): void

+ dragOver(DragEvent): void

+ dragDropped(DragEvent): void

+ dragOverBin(DragEvent): void

+ dragDroppedBin(DragEvent): void

+ dragDetectedArrow(MouseEvent): void

+ dragDetectedBomb(MouseEvent): void

+ dragDetectedBoulder(MouseEvent) void

+ dragDetectedCoward(MouseEvent): void

+ dragDetectedDoor(MouseEvent): void

+ dragDetectedExit(MouseEvent): void

+ dragDetectedHound(MouseEvent): void

+ dragDetectedHp(MouseEvent): void

+ dragDetectedHunter(MouseEvent): void

+ dragDetectedIp(MouseEvent): void

Player

bs: int

ws: int

rds: int

er: boolean

ncibleTime: int

s: ArrayList<Integer>

Bombs():int

Arrows():int

Swords():int

Bombs():int

InvincibleTime():int

pBomb(Board): boolean

---

1   1   1

Trap

Exit

Door

FloorSwitch

Boulder

FloorSwitch

Wall

BoulderSwitch

LitBomb

Item

Arrow

InvinciblityPotion

Sword

HoverPotion

Key

Treasure

UnlitBomb

1

<<enumeration>> **Direction**
+ directions(): ArrayList<Direction>

FlyingArrow

1

<<interface>> **MovementBehaviour**
+ nextDirection(Board, MovingEntity): Direction

PredictBehaviour

RunAwayBehaviour

MoveBetweenBehaviour

GoTowardsPlayer

Enemy

DoorProtector

Coward

Hound

Strategist

Hunter

+ sho
+ add
+ add
+ add
+ add
+ add
+ getl
+ isH
+ star
+ isln
+ has
+ disp
+ disp
+ upd

- board
- playe
- numl
- numT
- numS
+ getB
+ setB
+ drop
+ shoc
+ mov
+ getN
+ getN
+ getN

otArrows(Board): boolean

Bombs(int):void

Arrrows(int):void

Swords(int): void

InviciblieTime():void

Keys(int):void

Keys(): ArrayList<Integer>

over(): boolean

rtHover():void

vincible():boolean

Key(int):boolean

patchX(): int

patchY(): int

ate(Board): void

1

1

+ dragDetectedKey(MouseEvent): void

+ dragDetectedPit(MouseEvent): void

+ dragDetectedPlayer(MouseEvent): void

+ dragDetectedStrat(MouseEvent): void

+ dragDetectedSwitch(MouseEvent): void

+ dragDetectedSword(MouseEvent): void

+ dragDetectedTreasure(MouseEvent): void

+ dragDetectedWall(MouseEvent): void

GamePlayer

d: Board

er: Player

EnemyObjectives: int

TreasureObjectives: int

SwitchObjectives: int

Board(): String

Board(Board): board

Bomb(Board): boolean

otArrows(Board): boolean

ePlayer(String): boolean

NumEnemyObjectives(): int

NumTreasureObjectives(): int

NumSwitchObjectives(): int

BoardLoader

- filePath: String

+ getFilePath(): String

+ setFilePath(String): void

+ loadBoards(<ArrayList<Board>): void

- load(File): Board

BoardWriter

- filePath: String

+ getFilePath(): String

+ setFilePath(String):void

+ writeBoard(Board): void