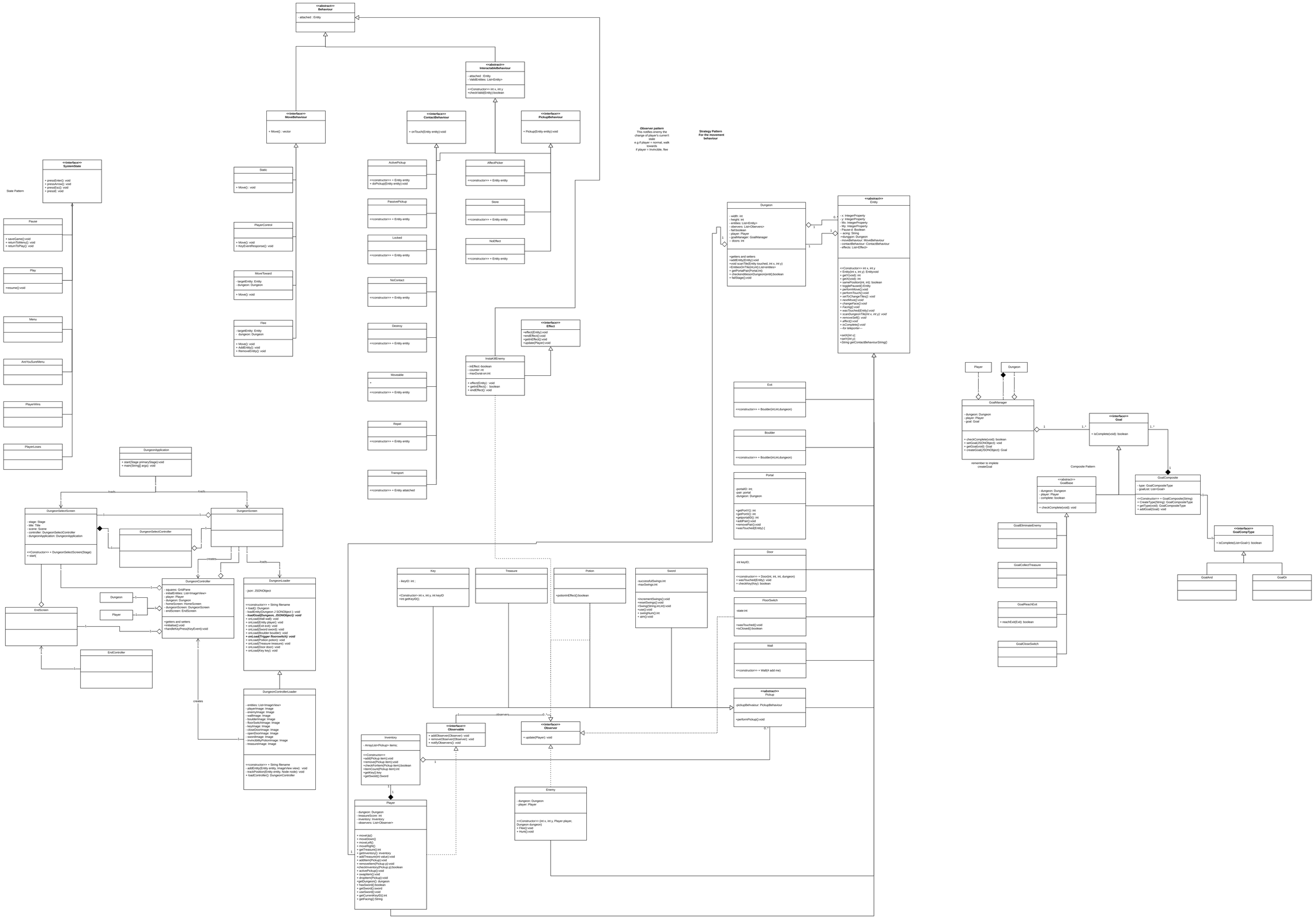
<<abstract>>
Behaviour
- attached : Entity
<<abstract>>
InteractableBehaviour
- attached : Entity
- ValidEntities: List<Entity>
<<Constructor>> int x, int y
+checkValid(Entity):boolean
<<interface>>
MoveBehaviour
+ Move() : vector
<<interface>>
ContactBehaviour
+ onTouch(Entity entity):void
<<interface>>
PickupBehaviour
+ Pickup(Entity entity):void
Observer pattern
This notifies enemy the change of player's current state
e.g if player = normal, walk towards
if player = invincible, flee
Strategy Pattern
For the movement behaviour
State Pattern
<<interface>>
SystemState
+ pressEnter(): void
+ pressArrow(): void
+ pressEsc(): void
+ pressE: void
Pause
+ saveGame():void
+ returnToMenu(): void
+ returnToPlay(): void
Play
+resume():void
Menu
AreYouSureMenu
PlayerWins
PlayerLoses
Static
+ Move() : void
PlayerControl
+ Move(): void
+ KeyEventResponse(): void
MoveToward
- targetEntity: Entity
- dungeon: Dungeon
+ Move(): void
Flee
- targetEntity: Entity
- dungeon: Dungeon
+ Move(): void
+ AddEntity(): void
+ RemoveEntity(): void
ActivePickup
<<constructor>> + Entity entity
+ doPickup(Entity entity):void
PassivePickup
<<constructor>> + Entity entity
Locked
<<constructor>> + Entity entity
NoContact
<<constructor>> + Entity entity
Destroy
<<constructor>> + Entity entity
Moveable
*
<<constructor>> + Entity entity
Repel
<<constructor>> + Entity entity
Transport
<<constructor>> + Entity attached
AffectPicker
<<constructor>> + Entity entity
Store
<<constructor>> + Entity entity
NotEffect
<<constructor>> + Entity entity
<<interface>>
Effect
+effect(Entity):void
+endEffect():void
+getinEffect():void
+updatePlayer():void
InstaKillEnemy
- inEffect:-boolean
- counter:-int
- maxDuration:int
+ effect()(Entity) : void
+ getinEffect() : boolean
+ endEffect(): void
Dungeon
- width: int
- height: int
- entities: List<Entity>
- observers: List<Observers>
- fail:boolean
- player: Player
- goalManager: GoalManager
- doors: int
+getters and setters
+addEntity(Entity):void
+void scanTile(Entity touched, int x, int y)
+ List<Entity> getEntities() +setentities>
+ getPortalPair(Portal,int)
+ checkendlesonDungeon(entit):boolean
+ failStage():void
<<abstract>>
Entity
- x: IntegerProperty
- y: IntegerProperty
- Mx: IntegerProperty
- My: IntegerProperty
- Pause-d: Boolean
- acing: String
-mdungon: Dungeon
- moveBehaviour: MoveBehaviour
- contactBehaviour: ContactBehaviour
- effects: List<Effect>
<<Constructor>> int x, int y
+ Entity(int x, int y): Entityvoid
+ getY(void): int
+ getX(void): int
+ samePosition(int, int): boolean
+ togglePaused():Entity
+ performMove():void
+ performTouch():void
+ setToChangeTiles(): void
+ changeFacing: void
+ changeFacing: void
+ Facing():void
+ wasTouched(Entity):void
+ scanDungeonTile(int x, int y): void
+ removeSelf(): void
+ affect():void
+ isComplete():void
--for teleporter--
+setX(int x);
+setY(int y);
+String getContactBehaviourString()
Exit
<<constructor>> + Boulder(intLint,dungeon)
Boulder
<<constructor>> + Boulder(intLint,dungeon)
Portal
-portalID: int;
-pair: portal
-dungeon: Dungeon
+getPortY(): int
+getPortX(): int
+getportalID(): int
+addPair():void
+removePair():void
+wasTouched(Entity) {
Door
-int keyID;
<<constructor>> + Door(int, int, int, dungeon)
+ wasTouched(Entity): void
+ checkKey(Key): boolean
FloorSwitch
-state:int
+wasTouched():void
+isClosed():boolean
Wall
<<constructor>> + Wall(if add me)
<<abstract>>
Pickup
pickupBehaviour: PickupBehaviour
+performPickup():void
Key
- keyID: int ;
<Constructor> int x, int y, int keyID
+int getKeyID();
Treasure
Potion
+potionInEffect():boolean
Sword
-successfulSwings:int
-maxSwings:int
+incrementSwings():void
+resetSwings():void
+Swing(String,int,int):void
+use():void
+ swingNum():int
+ aim():void
<<interface>>
Observable
+ addObserver(Observer): void
+ removeObserver(Observer): void
+ notifyObservers(): void
<<interface>>
Observer
+ update(Player): void
Inventory
- ArrayList<Pickup> items;
<<Constructor>>
+add(Pickup item):void
+remove(Pickup item):void
+checkFortem(Pickup item):boolean
+ItemCount(Pickup item):int
+getKey(): key
+getSword():Sword
Player
- dungeon: Dungeon
- treasureScore: int
- inventory: Inventory
- observers: List<Observer>
+ moveUp()
+ moveDown()
+ moveLeft()
+ moveRight()
+ getTreasure():int
+ getInventory(): Inventory
+ addTreasure(int value):void
+ addItem(Pickup):void
+ removeItem(Pickup p):void
+checkInventory(Pickup p):boolean
+ activePickup():void
+ swapItem():void
+ dropItem(Pickup):void
+ getDungeon(): Dungeon
+ hasSword():boolean
+ getSword():sword
+ useSword():void
+ getCurrentKeyID():int
+ getFacing():String
Enemy
- dungeon: Dungeon
- player: Player
<<Constructor>> (int x, int y, Player player, Dungeon dungeon)
+ Flee():void
+ Hunt():void
Player
Dungeon
GoalManager
- dungeon: Dungeon
- player: Player
- goal: Goal
+ checkComplete(void): boolean
+ setGoal(JSONObject): void
+ getGoal(void): Goal
+ createGoal(JSONObject): Goal
remember to implete
createGoal
<<interface>>
Goal
+ isComplete(void): boolean
Composite Pattern
<<abstract>>
GoalBase
- dungeon: Dungeon
- player: Player
- complete: boolean
+ checkComplete(void): void
GoalComposite
- type: GoalCompositeType
- goalList: List<Goal>
<<Constructor>> + GoalComposite(String)
+ CreateType(String): GoalCompositeType
+ getType(void): GoalCompositeType
+ addGoal(Goal): void
<<interface>>
GoalCompType
+ isComplete(List<Goal>): boolean
GoalEliminateEnemy
GoalCollectTreasure
GoalReachExit
+ reachExit(Exit): boolean
GoalCloseSwitch
GoalAnd
GoalOr
DungeonApplication
+ start(Stage primaryStage):void
+ main(String[] args): void
loads
DungeonSelectScreen
- stage: Stage
- title: Title
- scene: Scene
- controller: DungeonSelectController
-dungeonApplication: DungeonApplication
<<Constructor>> + DungeonSelectScreen(Stage)
+ start()
DungeonSelectController
DungeonScreen
creates
DungeonController
- squares: GridPlane
- initialEntities: List<ImageView>
- player: Player
-dungeon: Dungeon
- homeScreen: HomeScreen
- dungeonScreen: DungeonScreen
- endScreen: EndScreen
+getters and setters
+initialize():void
+handleKeyPress(KeyEvent):void
DungeonLoader
- json: JSONObject
<<constructor>> + String filename
+ load(): Dungeon
- loadEntity(Dungeon J,JSONObject): void
- **loadPlayer(Dungeon d,JSONObject): void**
+ onLoad(Wall wall): void
+ onLoad(Entity player): void
+ onLoad(Exit exit): void
+ onLoad(Sword sword): void
+ onLoad(Boulder boulder): void
+ ***onLoad(Trigger floorswitch): void***
+ onLoad(Potion potion): void
+ onLoad(Treasure treasure): void
+ onLoad(Door door): void
+ onLoad(Key key): void
Dungeon
Player
EndScreen
EndController
DungeonControllerLoader
- entities: List<ImageView>
- playerImage: Image
- enemyImage: Image
- wallImage: Image
- boulderImage: Image
- floorSwitchImage: Image
- keyImage: Image
- closeDoorImage: Image
- openDoorImage: Image
- swordImage: Image
- invincibilityPotionImage: Image
-treasureImage: Image
<<constructor>> + String filename
- addEntity(Entity entity, ImageView view): void
- trackPosition(Entity entity, Node node): void
+ loadController(): DungeonController