Game

- entityManager: EntityManager
- goal: GoalComponent

+ start()
+ pause()
+ resume()
+ restart()

MoveableComponent

- weight: int

CollectorComponent

- sword: Sword
- weapons: Set<IWeapon>
-keys: Set<IKey>

+ pickupSword(CollectableComponent<Sword>): boolean
+ pickupWeapon(CollectableComponent<IWeapon>): boolean
+ pickupKey(CollectableComponent<IKey>): boolean
+ attack(HealthComponent): void
+ unlock(LockComponent): boolean

Teleporter

- dest: Point2D
- destId: int

+ teleport(Entit

CollectableComponent

- collectable: T

+ get(): T

FollowComponent

- target: Entity

+ follow(Entity): void
+ nextMove(): void

Component

entity: Entity

LockComponent

- keyId: int
- status: LockState

+ unlock(int): boolean
+ isUnlocked(): boolean

MoverCo

- strength: int

ZoneComp

- isTriggered: boolean

- onEnter(entity: Entity, Zon
- onLeave(entity: Entity, Zor

<<enumeration>>
LockState

Locked
Unlocked

PlayerInputComponent

- moveEvents: IObservable<MoveEvent>

- handleMoveEvent(MoveEvent): void

ObservableComponent

- observable: IObservable<T>

+ addObserver (IObserver<T>): void
+ removeObserver(IObserver<T>): void
+notifyObserver(T): void

TransformComponent

position: Point2D

+ translate(dx: int, dy: int): void
+ translateX(dx: int): void
+ translateY(dy: int): void
+ isAt(x: int, y: int): boolean
+ equals(Object): boolean

HealthComponent

- hits: int

+ getHits(): int
+ hit(damage: int): void
+ die(): void

Extends (relationship labels throughout diagram)

MoveEvent

+ direction: Direction

+ up(): MoveEvent
+ right(): MoveEvent
+ down(): MoveEvent+ left():

<<enumeration>> Direction

UP
RIGHT
DOWN
LEFT

<<Interface>> IObserver

+ update(T): void

---

rComponent

y): void

Entity

- components: Set<Component>
- name: String

+ getComponent(type: Class): Component
+ addComponent(component: Component):
Component
+ removeComponent(type: Class): void
+addCollisionStrategy(entity: Entity,
strategy: CollisionStrategy)

Delta

+ from: T
+ to: T

<<Interface>> CollisionStrategy

+ exec(Delta<Point2D>): void

---

omponent

---

onent

eHandler): void
neHandler): void

Point2D

- x: int
- y: int

+ equals(Object): boolean

EntityFactory

+ BOULDER_WEIGHT: int
+ SWORD_DAMANGE: int
+ SWORD_NHITS : int
+TREASURE_VALUE: int
+ INVINC_DURATION_MS: int
+ UID: int

 - withUID(id: String): String
+ createPlayer(int, int, IObservable): Entity
+ createWall(int, int): Entity
+ createExit(int, int): Entity
+ createTreasure(int, int): Entity
+ createDoor(int, int, int): Entity
+ createKey(int, int, int): Entity
+ createBoulder(int, int): Entity
+ createFloorSwitch(int, int): Entity
+ createPortal(int ,int, int): Entity
+ createEnemy(int, int): Entity
+ createSword(int, int): Entity
+ createInvincibilityPotion(int, int): Entity

Treasure

- value: int

DoorKey

- keyId: int

+ unlock(LockComponent): boolean

<<Interface>>
IKey

+ unlock(LockComponent): boolean

<<Interface>>
IObservable

+ addObserver(IObserver<T>): v
+ removeObserver(IObserver<T
+ notifyObservers(T): void

Extends

Extends

InvincPotion

- activeFor: int
- isActive: boolean

+ attack(HealthComponent): void
+ activate(): void
+ update(dt: int)

<<Interface>>
IWeapon

+ attack(HealthComponent): void
+ activate() : void

Sword

- damage: int
- nhits: int

+ attack(HealthComponent): void

Extends

Extends

Extends

Observable

- observers: Set<IObserver<T>>

+ addObserver(IObserver<T>): void
+ removeObserver(IObserver<T>): void
+ notifyObservers(T): void

void
>): void