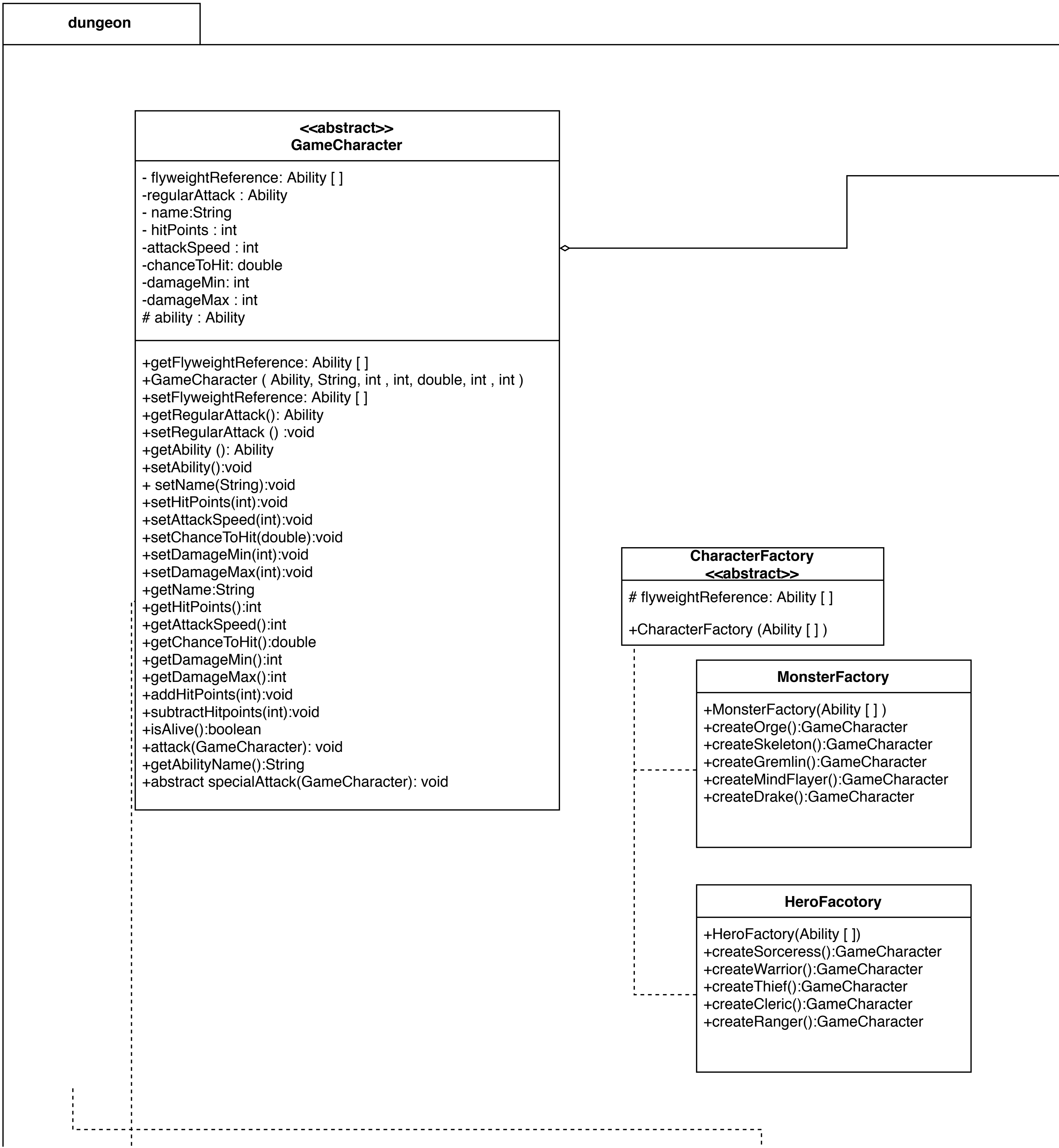


dungeon

<<abstract>>
GameCharacter

- flyweightReference: Ability [ ]
-regularAttack : Ability
- name:String
- hitPoints : int
-attackSpeed : int
-chanceToHit: double
-damageMin: int
-damageMax : int
# ability : Ability

+getFlyweightReference: Ability [ ]
+GameCharacter (Ability, String, int , int, double, int , int)
+setFlyweightReference: Ability [ ]
+getRegularAttack(): Ability
+setRegularAttack () :void
+getAbility (): Ability
+setAbility():void
+ setName(String):void
+setHitPoints(int):void
+setAttackSpeed(int):void
+setChanceToHit(double):void
+setDamageMin(int):void
+setDamageMax(int):void
+getName:String
+getHitPoints():int
+getAttackSpeed():int
+getChanceToHit():double
+getDamageMin():int
+getDamageMax():int
+addHitPoints(int):void
+subtractHitpoints(int):void
+isAlive():boolean
+attack(GameCharacter): void
+getAbilityName():String
+abstract specialAttack(GameCharacter): void

CharacterFactory
<<abstract>>

# flyweightReference: Ability [ ]

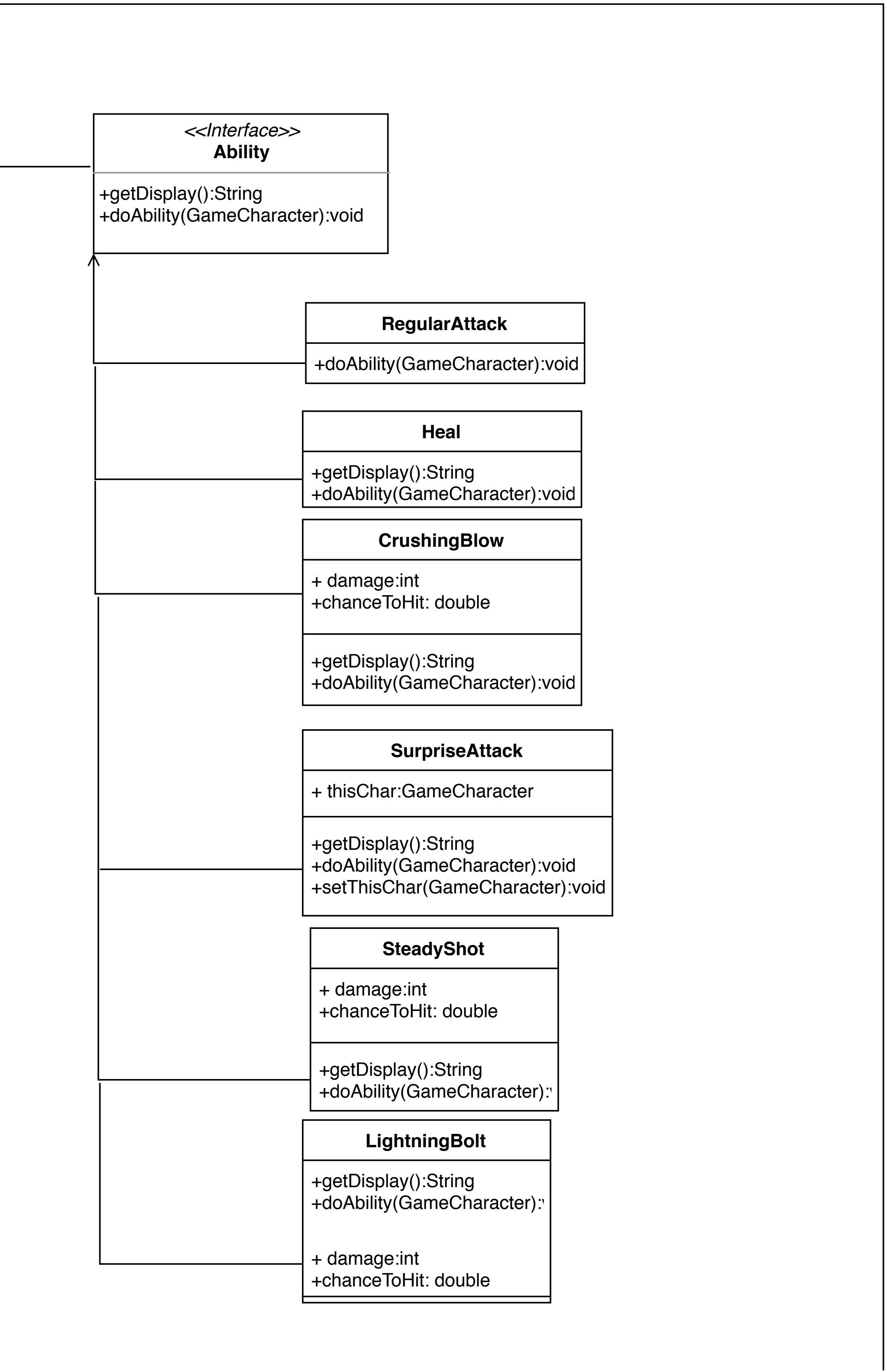
+CharacterFactory (Ability [ ])

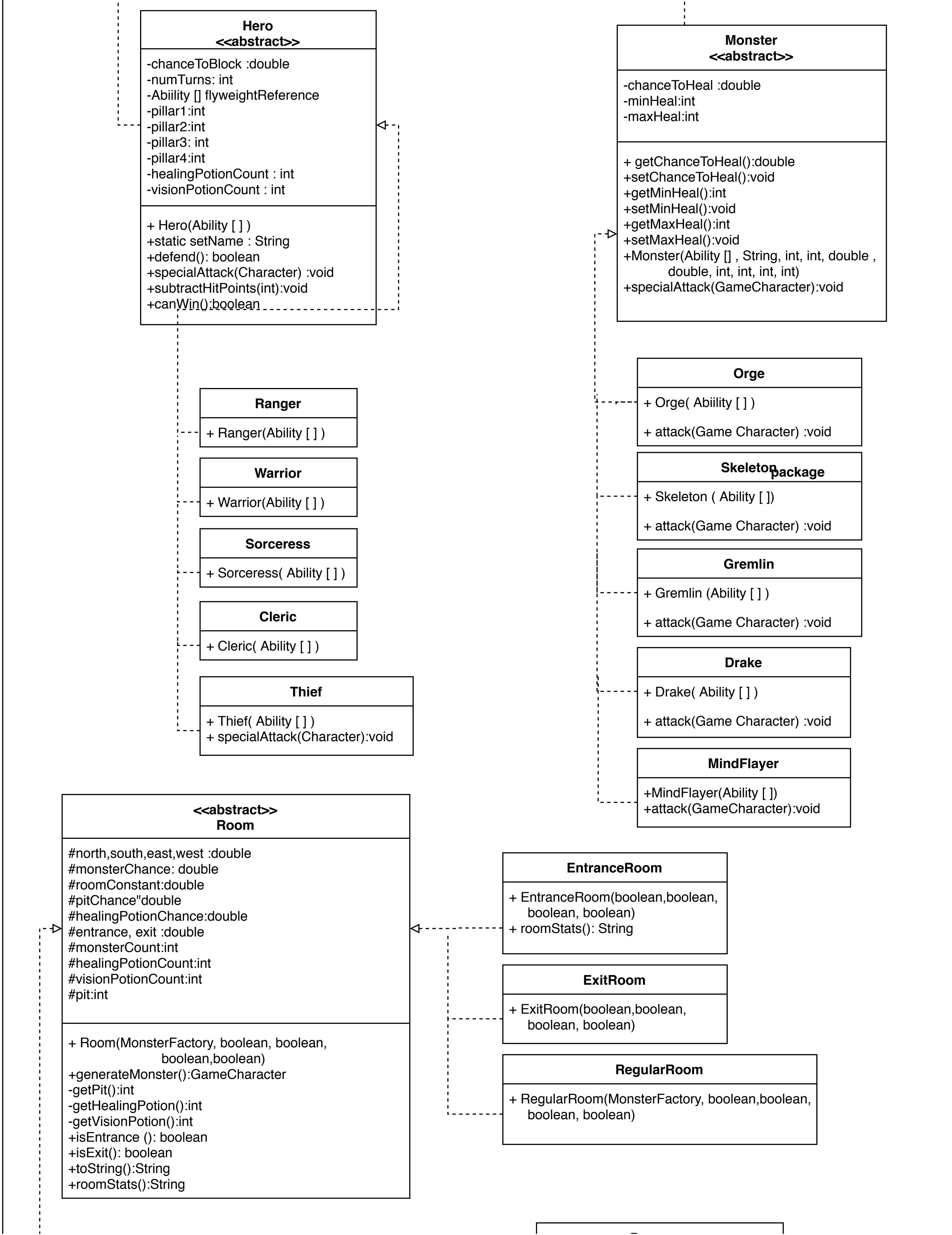
MonsterFactory

+MonsterFactory(Ability [ ])
+createOrge():GameCharacter
+createSkeleton():GameCharacter
+createGremlin():GameCharacter
+createMindFlayer():GameCharacter
+createDrake():GameCharacter

HeroFacotory

+HeroFactory(Ability [ ])
+createSorceress():GameCharacter
+createWarrior():GameCharacter
+createThief():GameCharacter
+createCleric():GameCharacter
+createRanger():GameCharacter


<<Interface>>
Ability
+getDisplay():String
+doAbility(GameCharacter):void
RegularAttack
+doAbility(GameCharacter):void
Heal
+getDisplay():String
+doAbility(GameCharacter):void
CrushingBlow
+ damage:int
+chanceToHit: double
+getDisplay():String
+doAbility(GameCharacter):void
SurpriseAttack
+ thisChar:GameCharacter
+getDisplay():String
+doAbility(GameCharacter):void
+setThisChar(GameCharacter):void
SteadyShot
+ damage:int
+chanceToHit: double
+getDisplay():String
+doAbility(GameCharacter):
LightningBolt
+getDisplay():String
+doAbility(GameCharacter):
+ damage:int
+chanceToHit: double



Hero
<<abstract>>

-chanceToBlock :double
-numTurns: int
-Abiility [] flyweightReference
-pillar1:int
-pillar2:int
-pillar3: int
-pillar4:int
-healingPotionCount : int
-visionPotionCount : int

+ Hero(Ability [ ])
+static setName : String
+defend(): boolean
+specialAttack(Character) :void
+subtractHitPoints(int):void
+canWin():boolean

Ranger

+ Ranger(Ability [ ])

Warrior

+ Warrior(Ability [ ])

Sorceress

+ Sorceress(Ability [ ])

Cleric

+ Cleric(Ability [ ])

Thief

+ Thief(Ability [ ])
+ specialAttack(Character):void

Monster
<<abstract>>

-chanceToHeal :double
-minHeal:int
-maxHeal:int

+ getChanceToHeal():double
+setChanceToHeal():void
+getMinHeal():int
+setMinHeal():void
+getMaxHeal():int
+setMaxHeal():void
+Monster(Ability [] , String, int, int, double ,
double, int, int, int, int)
+specialAttack(GameCharacter):void

Orge

+ Orge(Abiility [ ])

+ attack(Game Character) :void

Skeletonpackage

+ Skeleton (Ability [ ])

+ attack(Game Character) :void

Gremlin

+ Gremlin (Ability [ ])

+ attack(Game Character) :void

Drake

+ Drake(Ability [ ])

+ attack(Game Character) :void

MindFlayer

+MindFlayer(Ability [ ])
+attack(GameCharacter):void

<<abstract>>
Room

#north,south,east,west :double
#monsterChance: double
#roomConstant:double
#pitChance"double
#healingPotionChance:double
#entrance, exit :double
#monsterCount:int
#healingPotionCount:int
#visionPotionCount:int
#pit:int

+ Room(MonsterFactory, boolean, boolean,
boolean,boolean)
+generateMonster():GameCharacter
-getPit():int
-getHealingPotion():int
-getVisionPotion():int
+isEntrance (): boolean
+isExit(): boolean
+toString():String
+roomStats():String

EntranceRoom

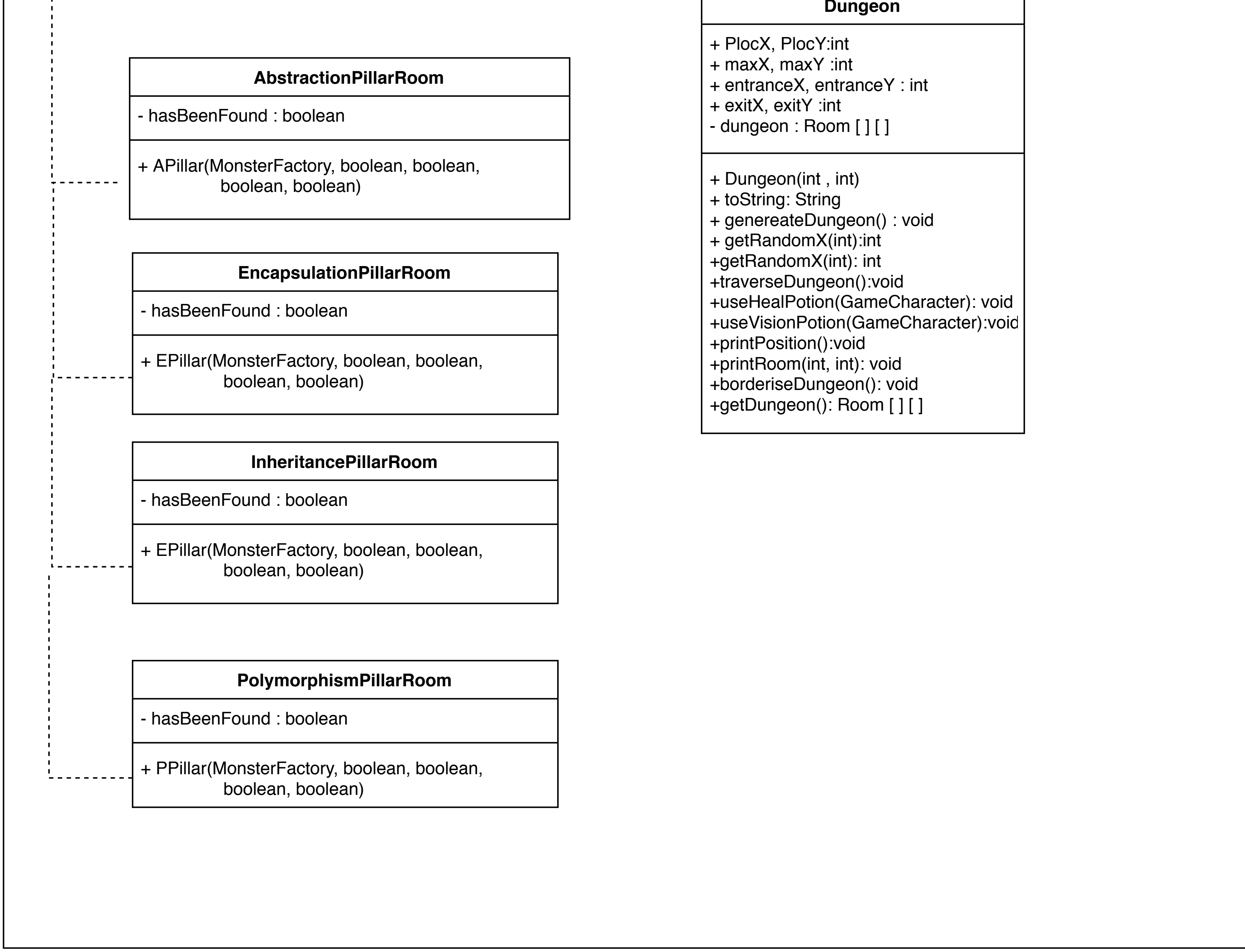
+ EntranceRoom(boolean,boolean,
boolean, boolean)
+ roomStats(): String

ExitRoom

+ ExitRoom(boolean,boolean,
boolean, boolean)

RegularRoom

+ RegularRoom(MonsterFactory, boolean,boolean,
boolean, boolean)



AbstractionPillarRoom

- hasBeenFound : boolean

+ APillar(MonsterFactory, boolean, boolean,
boolean, boolean)

EncapsulationPillarRoom

- hasBeenFound : boolean

+ EPillar(MonsterFactory, boolean, boolean,
boolean, boolean)

InheritancePillarRoom

- hasBeenFound : boolean

+ EPillar(MonsterFactory, boolean, boolean,
boolean, boolean)

PolymorphismPillarRoom

- hasBeenFound : boolean

+ PPillar(MonsterFactory, boolean, boolean,
boolean, boolean)

Dungeon

+ PlocX, PlocY:int
+ maxX, maxY :int
+ entranceX, entranceY : int
+ exitX, exitY :int
- dungeon : Room [ ] [ ]

+ Dungeon(int , int)
+ toString: String
+ genereateDungeon() : void
+ getRandomX(int):int
+getRandomX(int): int
+traverseDungeon():void
+useHealPotion(GameCharacter): void
+useVisionPotion(GameCharacter):void
+printPosition():void
+printRoom(int, int): void
+borderiseDungeon(): void
+getDungeon(): Room [ ] [ ]