<<enumeration>> Weapon

KATANA
BROADSWORD
TWOHANDED
AXE
FLAIL
NOWEAPON

+ toString(): String

<<interface>> Gear

+ getName(): String
+ getAbility(): Ability
+ getValue(): int
+ getSize(): int
+ void setPoison(boolean b)
+ boolean isPoisoned(): boolean

<<interface>> Battle

getPlayer(int playerId): Player
getCurrentTurnPlayerId(): int
getWinner(): Player
equipGears(int playerId)
requestWeapon(int playerId)
attack(): int
reset()

PlayerImpl

- name: String
- strength: int
- constitution - int
- dexterity: int
- charisma: int
- weapon: Weapon
- gearSet: List<Gear>
- totalDamage: int

+ getName(): String
+ getStrength(): int
+ getConstitution(): int
+ getDexterity(): int
+ getCharisma(): int
+ getWeapon(): Weapon
+ equipGear(Gear gear):
+ getGearList(): List<Gear>
+ getHealth(): int
+ getStrikingPower(): int
+ getAvoidanceAbility(): int
+ getPotentialStrikingDamage(): int
+ getTotalDamage(): int
+ attack(Player player): int
+ isDefeated(): boolean:
+ getRemainingHealth(): int

BattleImpl

- player: Player[]
- winner: Player
- gearBag: List<Gear>

+ BattleImpl(String player1Name, String player2Name, List<Gear> gearBag)

AbstractGear

name: String
ability: Ability
value: int
poisoned: boolean

~ AbstractGear(String name, Ability ability, int value)
# compareToBelt(Belt belt): int
# compareToHeadGear(HeadGear headGear): int
# compareToFootwear(Footwear footwear): int
# compareToPotion(Potion potion): int
+ compareTo(Gear o): int

HeadGear

index: 0

+ HeadGear(String name, int value)
# compareToHeadGear (HeadGear headGear): int
- compareTo(Gear o): int

Potion

index: 1

+ Potion(String name, Ability ability, int value)
# compareToPotion(Potion potion): int
- compareTo(Gear o): int

Belt

index: 2
beltSize: BeltSize

+ Belt(String name, BeltSize beltSize, Ability ability, int value)
# getSize(): int
# compareToBelt(Belt belt): int
- compareTo(Gear o): int

Footwear

index: 3

+ Footwear(String name, int value)
# compareToFootwear (Footwear footwear): int
- compareTo(Gear o): int

<<enumeration>> BeltSize

+ operation1(params):returnType
- operation2(params)
- operation3()

0..*
0..1
1
1
1
1
2
1
1

Testing plan will require to check all the conditions for the following method tests:

For BattleImpl:

testConstructor()

testGetPlayer()

testGetTurn()

testGetWinner()

testEquipGears()

testRequestWeapon()

testAttack()

For Player:

testConstructor()

testGetName()

testGetStrength()

testGetConstitution()

testGetDexterity()

testGetCharisma()

testEquipWeapon()

testEquipGear()

testGetHealth()

testGetStrikingPower()

testGetAvoidanceAbility()

testGetPotentialStrikingDamage()

testGetTotalDamage()

testIsDefeated()

testGetGearList()

For AbstractGear:

testConstructor()

testGetName()

testGetAbility()

testGetValue()

testPoisoned()

testHeadGearCompareTo()

testPotionCompareTo()

testBeltCompareTo()

testFootwearCompareTo()

testSortGears()