Our model is separate from controller and view. It can handle the corresponding situation and player actions. Thus, we only add <<interface>> BuildWorld ReadOnlyWorld interface and implement getters for View to read information needed in the model. + getRoomList() : List<Room> + getTargetCharacter() : Piece + getPlayerList() : List<Player> getMaxTurn(): int getCurrentPlayer(turn : int) : Player getLocation(player : Player) : Space getItemInRoom(id : int) : List<Item> + getNeigList(int id) : List<Space> <<interface>> + lookAround(playerId : int) : String WorldController + displayRoomInfo(playerId : int) : String + displayPlayerInfo(playerId: int):String generateMap(): void + moveTarget() : String + move(playerId : int, roomId : int) : String + pickItem(playerId : int, itemId : int) : String addPlayer(name : String, id : int, isHuman : boolean, bagCapacity: int, roomId: int <<interface>> + randomNum(max : int) : int ReadOnlyWorld + attackTarget(playerId : int, itemId : int) : String + pokeTarget(playerId : int) : String + getPlayerList():List<Player> + movePet(roomId : int) : String + getRoomList(): List<Space> + getPet() : Pet <<interface>> + findRoomId(x : int, y : int) : int + targetHealth(): int + hasItem(playerId : int) : Boolean WorldView + displayItemInBag(playerId : int) :String + getMap() : BufferedImage + canBeSeen(player : Player) : boolean + computerAction(playerId : int) : String + movePetFollowDfs(startRoomId : int, + randomNum(max : int) : int numMoves: int)::void + getAllRoomsName() : String + computerCmd(playerId : int) : String + isGameOver : Boolean + findRoomId(x : int, y : int) : int + displayItemInRoom(playerId : int) : String + getTurn() : int + nextTurn() : void + displayGameStates(playerId : String) : String + setMaxTurn(max : int) + getFilePath() : String + findRoomByPlayerId(playerId: int): Space BuildWorldModel image : BiufferedImage <<interface>> - mapCol : int - mapRow : int - numRoom : int + move(roomId: int): void numWeapons: int + getName() : String - tempRoom : Room + getCurrentRoomId(): int - roomList: List<Room> - targetName : String - targetHealth : int - targetCharacter : Piece - playerList: List<Player> - pet: Pet petName : String - winner : String TargetPet name: String + BuidWorldModel(args : String[], turns : int) - currentRoomId : int - checkOverlappedRoom() : void <<interface>> - isOverlapped(room1 : Space, room2 : Space) : <<interface>> **Space** boolean Player + TargetPet(name : String) - findNeighbors(): void + move(roomId : int) : void + getRoomList() : List<Room> + getName() : String + getName() : String getTargetCharacter(): Piece + getCurrentRoomId(): int + move(roomId:int) : void + getItems() : List<item> + getPlayerList() : List<Player> + equals() : boolean + getName() : String + getUpRow() : int + getMaxTurn() : int + hashCode(): int + getUpCol() : int + getId(): int getCurrentPlayer(turn : int) : Player + getDownRow() : int + getCurrentRoomId(): int getLocation(player : Player) : Space + getDownCol(): int + pickItem(item:Item):void getItemInRoom(id : int) : List<Item> + getId() : int + getItemList():List<Item> + getNeigList(int id) : List<Space> + getIsHuman():boolean + getNeighbors() : List<Space> + lookAround(playerId : int) : String + getBagCapacity():int + addItem(item: Item) : void + displayRoomInfo(playerId : int) : String + removeItem(item: Item) : void + removeItem(item : Item):void + displayPlayerInfo(playerId : int):String + isNeigjhbor(room: Space) : void + generateMap() : void + getItemList() : List<Item> + moveTarget() : String + getPlayers() : List<Player> + move(playerId : int, roomId : int) : String + addPlayer(Player player) : void + pickItem(playerId : int, itemId : int) : String + removePlayer(Player player) : void + addPlayer(name : String, id : int, isHuman : <<interface>> <<interface>> boolean, bagCapacity: int, roomId: int) Item Piece + randomNum(max : int) : int + attackTarget(playerId : int, itemId : int) : String PlayerCharacter + move() : void + pokeTarget(playerId : int) : String + getName() : String + getHealth() : int + movePet(roomId : int) : String - currentRoomId : int + getPower() : int getName() : String + getPet() : Pet - name : String - getCurrentRoomId() : int + targetHealth(): int - id : int reduceHealth(damage : int) : void + displayItemInBag(playerId : int) :String - itemsInBag : List<Item> + canBeSeen(player : Player) : boolean Room - isHuman: boolean + movePetFollowDfs(startRoomId : int, - name : String numMoves : int) : void + PlayerCharacter(name:String, id:int, - upCol : int - dfsPath(startRoomId : int) : List<Integer> isHuman: boolean, bagCapactity: int, - upRow: int + computerCmd(playerId : int) : String currentRoomId :int) - downCol: int + isGameOver : Boolean + move(roomId:int) : void - downRow: int + findRoomId(x : int, y : int) : int + getName() : String - id: int + displayItemInRoom(playerId : int) : String getId(): int - items : List<Item> + getTurn() : int getCurrentRoomId(): int - neighbors : List<Space> + nextTurn() : void pickItem(item:Item):void - players: List<Player> + displayGameStates(playerId : String) : String + getItemList():List<Item> + setMaxTurn(max : int) + getIsHuman():boolean + getFilePath() : String + Room(iname : String, Id : id, upRow : int, + getBagCapacity():int TargetCharacter + findRoomByPlayerId(playerId : int) : Space id : int, upCol : int, downRow : int, + equals():boolean 0..* Weapon downCol : int) + hashCode():int - currentRoomId : int = 0 + getName() : String removeItem(item : Item):void - name : String - name : String + getItems() : List<item> - health : int sortItemList(List<Item>) : void - power : int + getUpRow() : int - id : numRoom + getUpCol() : int getDownRow(): int Weapon(name : String, power : int) + TargetCharacter(name: Sting, getDownCol(): int getPower(): int health: int, numRoom: int) + getId() : int + getName() : String + move() : void + getNeighbors() : List<Space> + equals() : boolean + addItem(item: Item) : void + getHealth(): int + hashCode(): int + removeItem(item: Item) : void + getName() : String + getCurrentRoomId() : int + isNeigjhbor(room : Space) : void + getItemList():List<Item> + equals(): boolean + hashCode(): int + getPlayers() : List<Player> + addPlayer(Player player) : void + reduceHealth(damage : int) : void + removePlayer(Player player) : void + equals() : boolean + hashCode(): int + toSting(): String

Changes that made the model to separate: