

Validator
- board
+ is_valid()
+ get_possible_moves

MoveValidator

BuildValidator

TurnManager
+ move_worker()
+ build()
+ next_turn()
+ check_win()

Game
- board: Board
- players: list(Player)
- turn_manager: TurnManager
- god_card_deck: List(GodCard)
- game_state: enum
+ initialize(width: int, height: int): void
+ setup_player(p1_name, p2_name): void
+ assign_god_card(): void
+ place_workers(): void
+ check_game_over(): bool



Player
- player_name: str
- player_age: int
- player_god: GodCard
- player_color: str(enum)
- player_workers: list(Worker)
+ get_god_card(): GodCard
+ choose_worker()

Worker
- worker_position: Position
- previous_position: Position
- previous_build_position: Position
+ get_position(): Position
+ set_position(): void
+ get_previous_position(): Position

Position
- x: int
- y: int
+ getX(): int
+ getY(): int

Board
- tiles: Tile(int)[int]
- width: int
- height: int
+ initialize(width: int, height: int): void
+ get_tile(position: Position): Tile
+

Tile
- position: Position
- worker: Optional[Worker]
- building: Optional[Building]
+ get_building(): Optional[Building]
+ has_worker(): bool

Building
- level: int(enum)
- dome: bool
+ has_dome(): bool
+ get_level(): int
+ increase_level(): bool



```

get_worker_level()
Tile = get_tile(x, y)
Worker Height = Tile.get_building.get_level()
(Level 1)

```

```

Check Valid Movement ->
Tile.get_building.get_level() (Level 3), compare
Level 3 - Level 1 == 2 So, not a valid move

```

```

if abs(new_row - current_row) > 1 or abs(new_col - current_col) > 1:
    return False

```

