Milestone 3 - Preliminary Design Testing Plan

Player

Method	Input	Expected Output
`canSee(PlayerInterface otherPlayer)`	`otherPlayer` in the same room	Returns `true`.
	`otherPlayer` in a visible neighboring room	Returns `true`.
	`otherPlayer` in a non-visible neighboring room	Returns `false`.
	`otherPlayer` in a room obscured by pet	Returns `false`.

Target

Method	Input	Expected Output
`getLastRoomVisited()`	Target was in Room A, moved to Room B	Returns `Room A`.
`setLastRoomVisited(RoomInterface lastRoom)`	Room C as `lastRoom`	Updates `lastRoomVisited` to Room C.
`reduceHealth(int damage)`	`damage = 5`	Reduces target's health by 5.
	`damage = 0`	No change to target's health.
`isAlive()`	Target health > 0	Returns `true`.
	Target health <= 0	Returns `false`.

Room

Method	Input	Expected Output
`getVisibleNeighbors()`	Room A with visible neighboring rooms B, C	Returns a list containing Room B and Room C.
`setSealed()`	Room is set to sealed	Room's sealed status becomes `true`.
`getSealed()`	Room previously set to sealed	Returns `true`.
	Room not sealed	Returns `false`.

Pet

Method	Input	Expected Output
`moveTo(RoomInterface room)`	Move pet to Room A	Pet's `currentRoom` is updated to Room A.
`isRoomVisible(RoomInterface room)`	Pet in Room A, check Room A visibility	Returns `false`.
	Pet in Room B, check Room A visibility	Returns `true`.
`wander()`	Pet wanders following depth-first traversal pattern	Pet moves to a neighboring room; returns confirmation of movement.

World

Method	Input	Expected Output
`attemptOnTarget(PlayerInterface player, ItemInterface item)`	Player in the same room as target, item with 5 damage	Returns `true`, reduces target's health by 5, removes item from inventory.
	Player in different room, item with 5 damage	Returns `false`, no change to target's health.
`movePet(RoomInterface room)`	Move pet to Room B	Pet's `currentRoom` is updated to Room B.
`checkGameEnd()`	Target health <= 0	Returns `true`, game ends.
	Max turns reached	Returns `true`, game ends.
`getVisiblePlayers(PlayerInterface player)`	Player in Room A with Player B visible	Returns a list containing Player B.
`getTargetLocationHint()`	Target in Room B, last room visited Room A	Returns a string such as "The target was last seen in Room A."

Controller

Method	Input	Expected Output
`handleCommand(String command)`	`command = "move pet to Room B"`	Executes pet movement to Room B.

	`command = "attempt on target with Sword"`	Executes attack attempt, returning result based on visibility conditions.
`checkVisibility(PlayerInterface player)`	Player in Room A with visible neighboring rooms B, C	Returns a list containing Room B and Room C.

AttemptOnTargetCommand

Method	Input	Expected Output
`execute()`	Player in same room as target, attack not visible	Attack is successful, target health decreases, item is removed from play.
	Player in same room as target, attack is visible	Attack is unsuccessful, no change to target's health, item is removed from play.

MovePetCommand

Method	Input	Expected Output
`execute()`	Command to move pet to Room B	Pet moves to Room B, `currentRoom` updated to Room B.

Conclusion

The updated UML and testing plan ensure all newly implemented features are covered for Milestone 3. Key additions include:

- **Room Trail Mechanic**: Tracking the target's last visited room for a realistic "chasing" experience, enhancing gameplay immersion.
- **Command Pattern for Modularity**: The command pattern enables a structured, flexible approach to actions such as moving the pet or attempting attacks on the target.
- **Pet Visibility Influence**: Introducing pet-related room visibility influences adds strategy, as players must account for the pet's location when planning attacks.

Each feature, especially those related to visibility and command pattern execution, has associated tests to verify both functionality and game rules. This testing plan allows for validating gameplay mechanics, ensuring a cohesive player experience aligned with the game's rules and objectives.