Awakening Capstone

JAVIER DURAN
DAMON (MENG) DENG
TIM ROBBINS

Awakening

ABOUT

Awakening Story

- Adventure / Escape themed game.
- Player wakes up in hospital with no knowledge of anything
- Interact with NPCs to determine how to escape

Awakening Story – ASCII to GUI journey

EXISTING BUGS

- Some puzzles not fully implemented
- Not all rooms initially accessible
- No defined win / lose scenarios

INCORPORATING SWING

- Determining how to work with existing framework
- Running the GUI client 'parallel' to existing game
- Finishing started game features and including our own

Learning / Challenges

ABOUT

Tim Robbins - Learning / Challenges

- Swing
  - Time frame to understand its capabilities and integrate into an unfamiliar code base
- Time Management / Managing backlog
  - Open dialogue within team is critical
  - Agile Principle 12: At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
- Working in an unfamiliar codebase
  - Importance of comments and adhering to good code practice

Damon - Learning / Challenges

- Working on inherited code bases
- Delivering product and features driven by the customer demands
  - Agile Principle 2: Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage...
- Building a virtual team
- Implementing best practices
- Documentation, comments

Javier – Learning / Challenges

- Interpret existing code
  - Needed time to understand the functionality of existing code
- Swing
  - Creating frames and integrating it to existing code
- Iteration Backlog
  - Creating user story tickets
- Test/Code Comments
  - Commenting new code and creating tests

Game Highlights

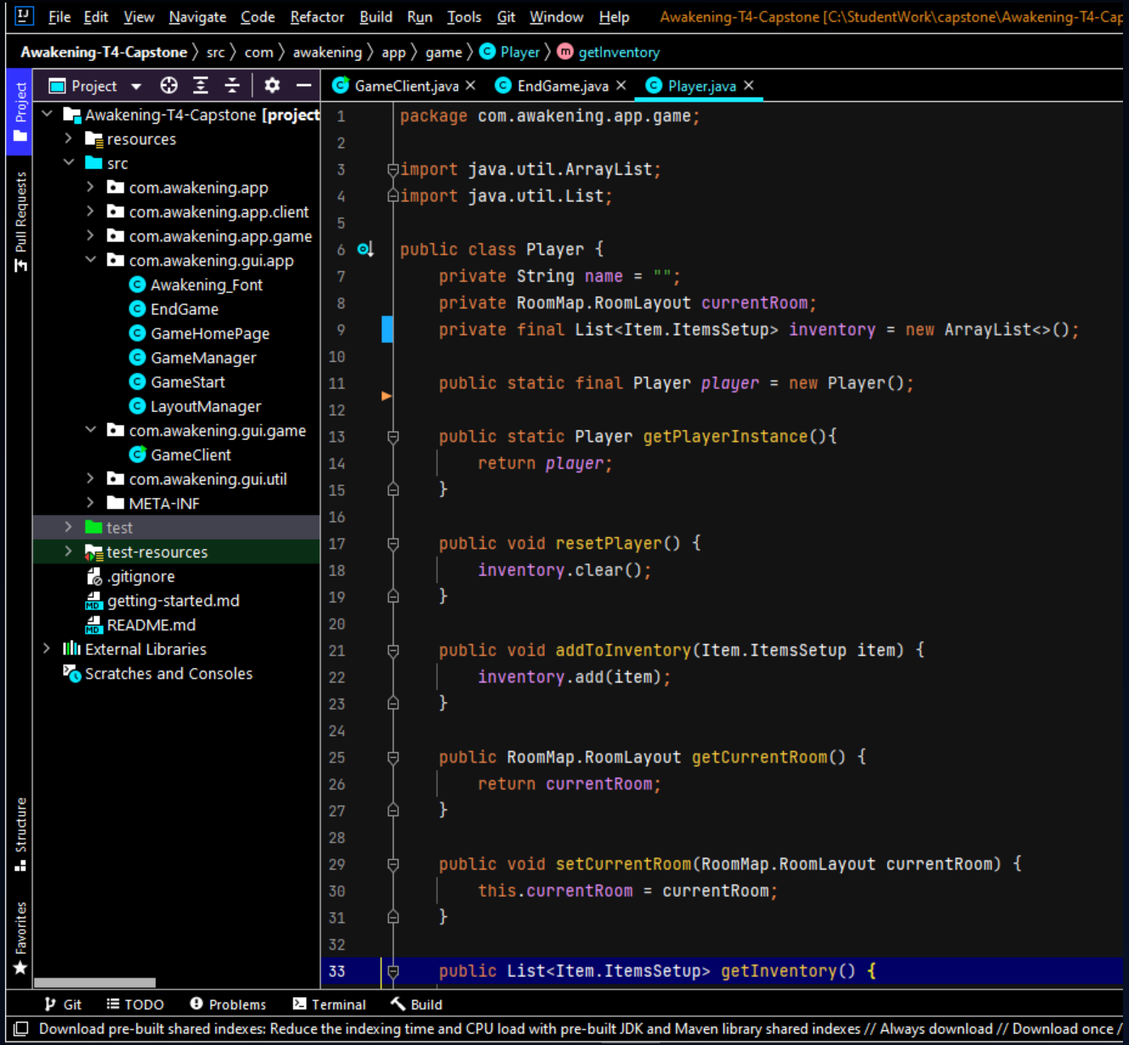
ABOUT

Game Highlights

- Light-weight text-based Retro adventure game
  - Solve challenging puzzles to escape the abandoned hospital while being chased by ghosts
- Solid Principle/Design Pattern
  - This helped us get more training and exposure to industry level best practices
- Swing GUI Implementation
  - With Swing we were able to provide a visual of the game which enhanced the user experience

Code Presentation

Damon - Code Sample



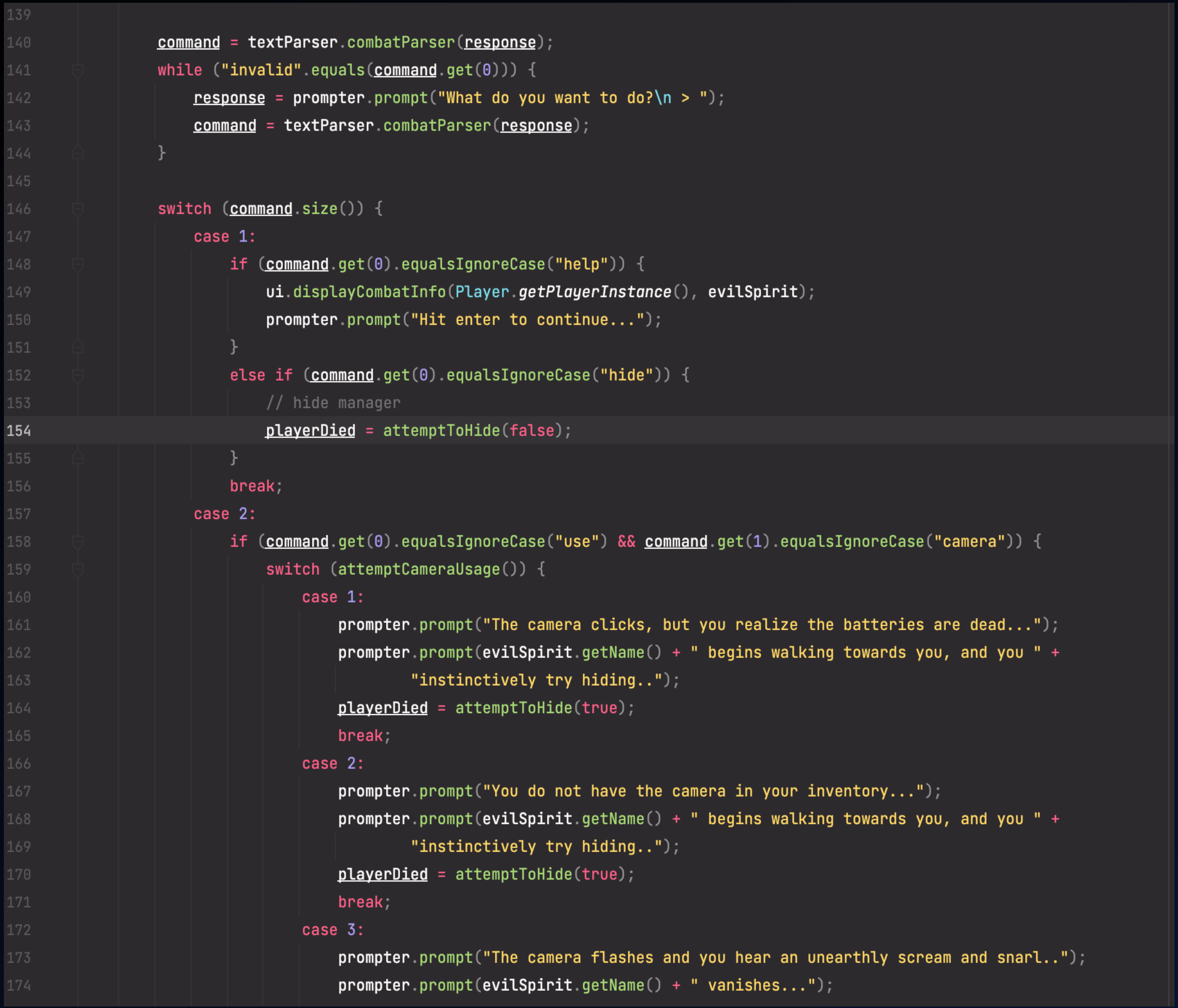
```java
package com.awakening.app.game;

import java.util.ArrayList;
import java.util.List;

public class Player {
    private String name = "";
    private RoomMap.RoomLayout currentRoom;
    private final List<Item.ItemsSetup> inventory = new ArrayList<>();

    public static final Player player = new Player();

    public static Player getPlayerInstance(){
        return player;
    }

    public void resetPlayer() {
        inventory.clear();
    }

    public void addToInventory(Item.ItemsSetup item) {
        inventory.add(item);
    }

    public RoomMap.RoomLayout getCurrentRoom() {
        return currentRoom;
    }

    public void setCurrentRoom(RoomMap.RoomLayout currentRoom) {
        this.currentRoom = currentRoom;
    }

    public List<Item.ItemsSetup> getInventory() {
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK and Maven library shared indexes // Always download // Download once /

Singleton design:

- Single player game:
  - Only **one** Player instance throughout the game
- Access control
  - Protects the class from unauthorized usages
- Prevents duplicates
  - Disable "new" keyword
- Up-to-date
  - Player's status, inventory, etc

Tim - Code Sample



```java
command = textParser.combatParser(response);
while ("invalid".equals(command.get(0))) {
    response = prompter.prompt("What do you want to do?\n > ");
    command = textParser.combatParser(response);
}

switch (command.size()) {
    case 1:
        if (command.get(0).equalsIgnoreCase("help")) {
            ui.displayCombatInfo(Player.getPlayerInstance(), evilSpirit);
            prompter.prompt("Hit enter to continue...");
        }
        else if (command.get(0).equalsIgnoreCase("hide")) {
            // hide manager
            playerDied = attemptToHide(false);
        }
        break;
    case 2:
        if (command.get(0).equalsIgnoreCase("use") && command.get(1).equalsIgnoreCase("camera")) {
            switch (attemptCameraUsage()) {
                case 1:
                    prompter.prompt("The camera clicks, but you realize the batteries are dead...");
                    prompter.prompt(evilSpirit.getName() + " begins walking towards you, and you " +
                            "instinctively try hiding..");
                    playerDied = attemptToHide(true);
                    break;
                case 2:
                    prompter.prompt("You do not have the camera in your inventory...");
                    prompter.prompt(evilSpirit.getName() + " begins walking towards you, and you " +
                            "instinctively try hiding..");
                    playerDied = attemptToHide(true);
                    break;
                case 3:
                    prompter.prompt("The camera flashes and you hear an unearthly scream and snarl..");
                    prompter.prompt(evilSpirit.getName() + " vanishes...");
```

Combat Engine

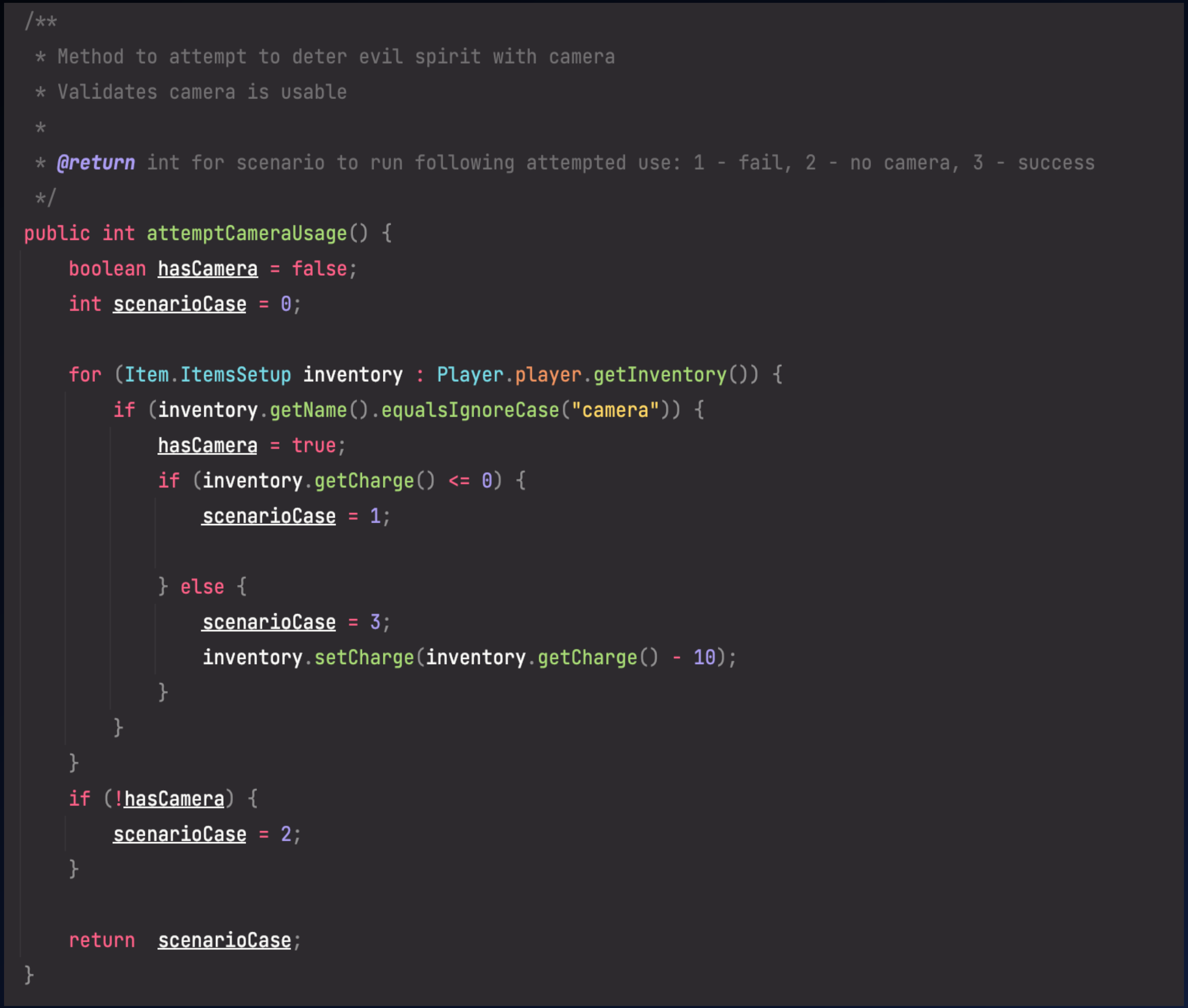
- Receives player command and pass to parser class for validation.
- Returned validated command is actioned based on command size
- Utilize helper method to determine game resp onse

Tim - Code Sample

Helper Method

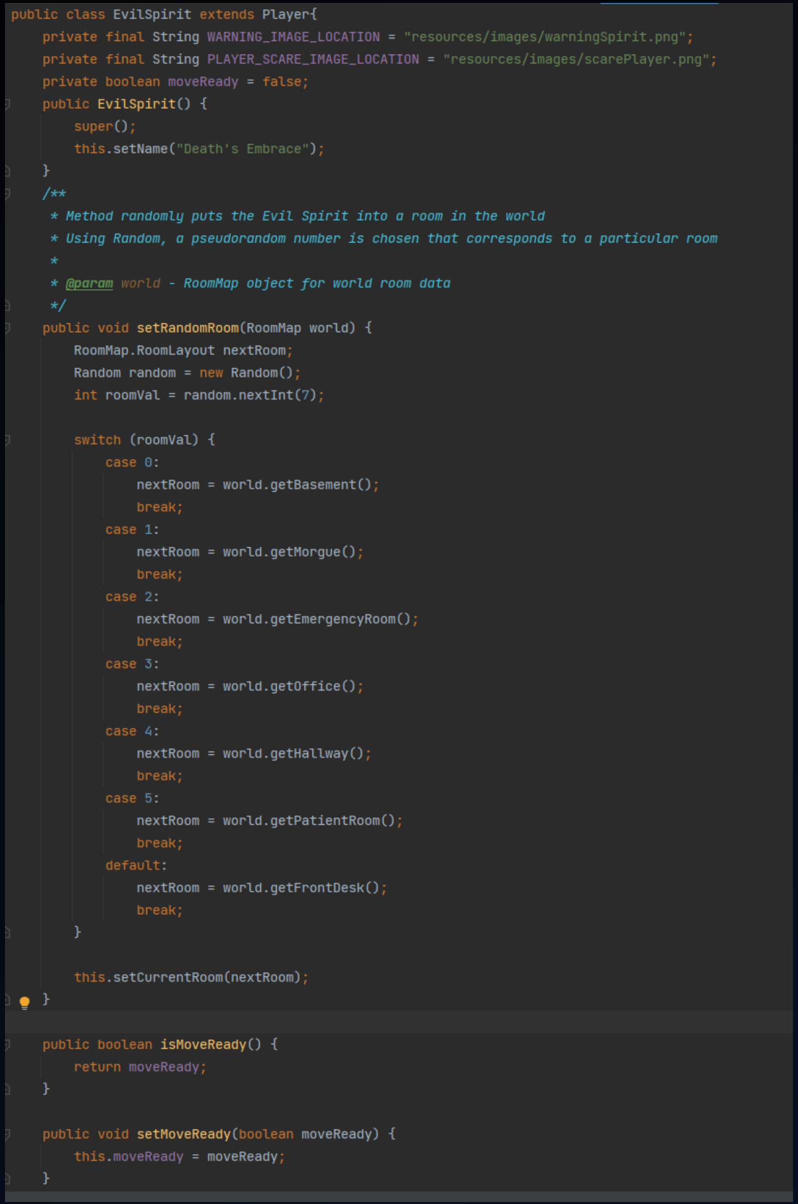
- Method assists combat engine with determining 'scenario' that the 'combat' ends in

- Returned integer is utilized in switch statement for driving game



```
/**
 * Method to attempt to deter evil spirit with camera
 * Validates camera is usable
 *
 * @return int for scenario to run following attempted use: 1 - fail, 2 - no camera, 3 - success
 */
public int attemptCameraUsage() {
    boolean hasCamera = false;
    int scenarioCase = 0;

    for (Item.ItemsSetup inventory : Player.player.getInventory()) {
        if (inventory.getName().equalsIgnoreCase("camera")) {
            hasCamera = true;
            if (inventory.getCharge() <= 0) {
                scenarioCase = 1;

            } else {
                scenarioCase = 3;
                inventory.setCharge(inventory.getCharge() - 10);
            }
        }
    }
    if (!hasCamera) {
        scenarioCase = 2;
    }

    return  scenarioCase;
}
```

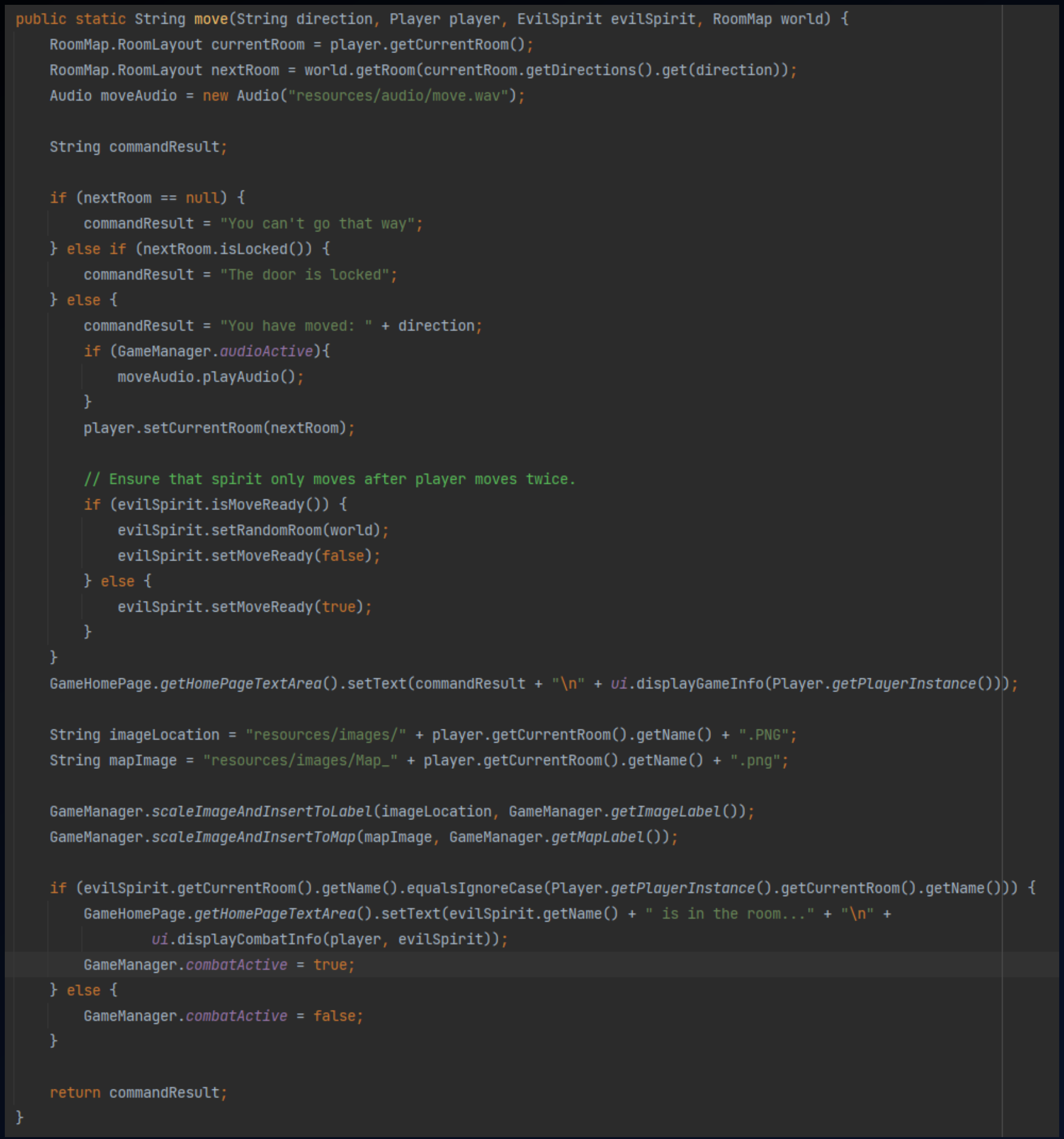
Javi – Code Sample



```java
public class EvilSpirit extends Player{
    private final String WARNING_IMAGE_LOCATION = "resources/images/warningSpirit.png";
    private final String PLAYER_SCARE_IMAGE_LOCATION = "resources/images/scarePlayer.png";
    private boolean moveReady = false;
    public EvilSpirit() {
        super();
        this.setName("Death's Embrace");
    }
    /**
     * Method randomly puts the Evil Spirit into a room in the world
     * Using Random, a pseudorandom number is chosen that corresponds to a particular room
     *
     * @param world - RoomMap object for world room data
     */
    public void setRandomRoom(RoomMap world) {
        RoomMap.RoomLayout nextRoom;
        Random random = new Random();
        int roomVal = random.nextInt(7);

        switch (roomVal) {
            case 0:
                nextRoom = world.getBasement();
                break;
            case 1:
                nextRoom = world.getMorgue();
                break;
            case 2:
                nextRoom = world.getEmergencyRoom();
                break;
            case 3:
                nextRoom = world.getOffice();
                break;
            case 4:
                nextRoom = world.getHallway();
                break;
            case 5:
                nextRoom = world.getPatientRoom();
                break;
            default:
                nextRoom = world.getFrontDesk();
                break;
        }

        this.setCurrentRoom(nextRoom);
    }

    public boolean isMoveReady() {
        return moveReady;
    }

    public void setMoveReady(boolean moveReady) {
        this.moveReady = moveReady;
    }
```

- EvilSpirit extends Player
- setRandomRoom()
  - When called, it provides a random number that corresponds to a room. Then sets the EvilSpirit to that room.
- isMoveReady()
- setMoveReady()

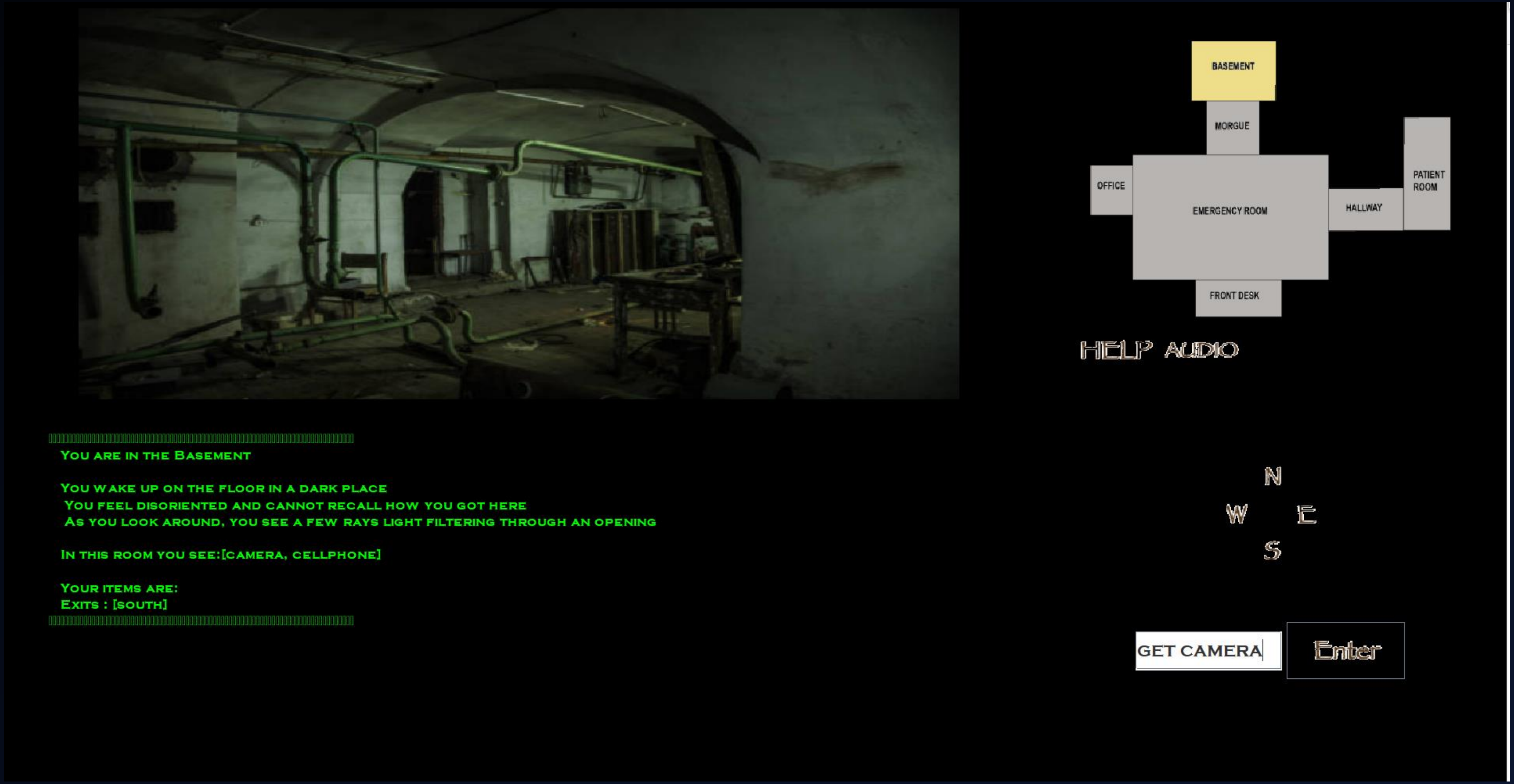
Javi – Code Sample



```java
public static String move(String direction, Player player, EvilSpirit evilSpirit, RoomMap world) {
    RoomMap.RoomLayout currentRoom = player.getCurrentRoom();
    RoomMap.RoomLayout nextRoom = world.getRoom(currentRoom.getDirections().get(direction));
    Audio moveAudio = new Audio("resources/audio/move.wav");

    String commandResult;

    if (nextRoom == null) {
        commandResult = "You can't go that way";
    } else if (nextRoom.isLocked()) {
        commandResult = "The door is locked";
    } else {
        commandResult = "You have moved: " + direction;
        if (GameManager.audioActive){
            moveAudio.playAudio();
        }
        player.setCurrentRoom(nextRoom);

        // Ensure that spirit only moves after player moves twice.
        if (evilSpirit.isMoveReady()) {
            evilSpirit.setRandomRoom(world);
            evilSpirit.setMoveReady(false);
        } else {
            evilSpirit.setMoveReady(true);
        }
    }
    GameHomePage.getHomePageTextArea().setText(commandResult + "\n" + ui.displayGameInfo(Player.getPlayerInstance()));

    String imageLocation = "resources/images/" + player.getCurrentRoom().getName() + ".PNG";
    String mapImage = "resources/images/Map_" + player.getCurrentRoom().getName() + ".png";

    GameManager.scaleImageAndInsertToLabel(imageLocation, GameManager.getImageLabel());
    GameManager.scaleImageAndInsertToMap(mapImage, GameManager.getMapLabel());

    if (evilSpirit.getCurrentRoom().getName().equalsIgnoreCase(Player.getPlayerInstance().getCurrentRoom().getName())) {
        GameHomePage.getHomePageTextArea().setText(evilSpirit.getName() + " is in the room..." + "\n" +
                ui.displayCombatInfo(player, evilSpirit));
        GameManager.combatActive = true;
    } else {
        GameManager.combatActive = false;
    }

    return commandResult;
}
```

- Move()
  - Checks if able to move to next room.
  - Changes GUI display
- isMoveReady()
  - Checks if EvilSpirit is ready to move location
- setMoveReady()
  - Makes changes after first move
- CombatActive
  - Is set when Player and EvilSpirit are in the same room.

Game Preview



```
  You are in the Basement

  You wake up on the floor in a dark place
   You feel disoriented and cannot recall how you got here
   As you look around, you see a few rays light filtering through an opening

  In this room you see:[camera, cellphone]

  Your items are:
  Exits : [south]
```