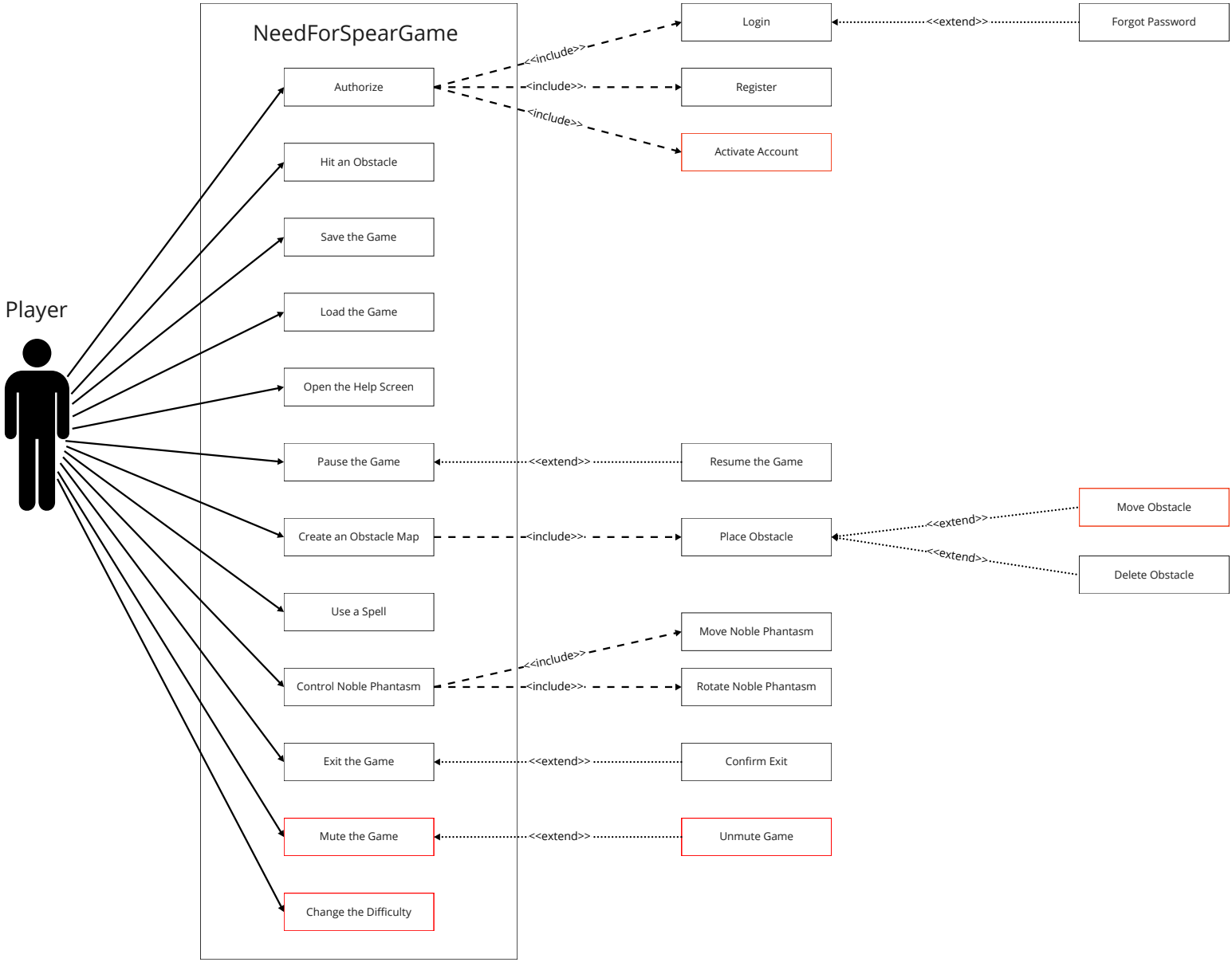
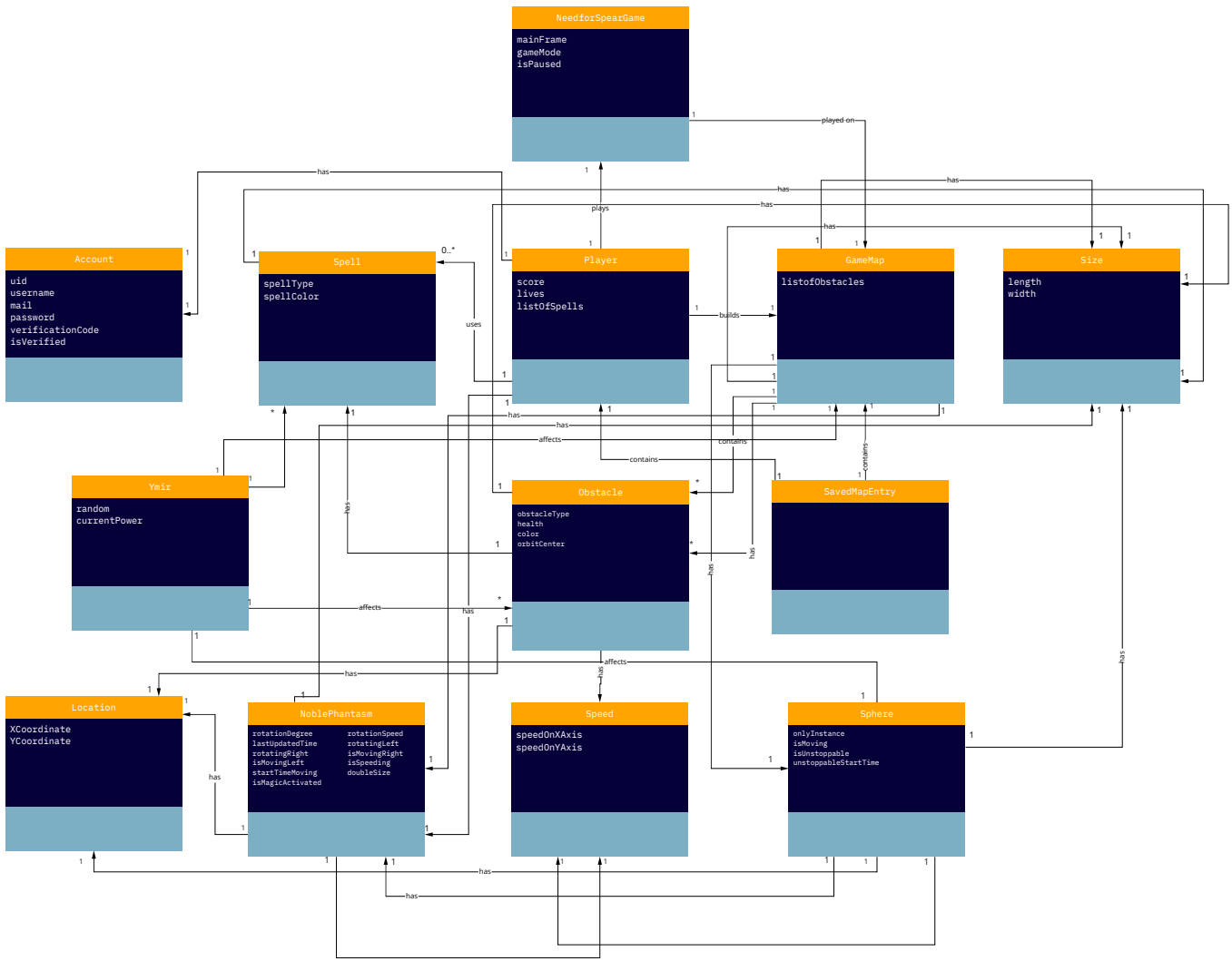
# Use Case Diagram

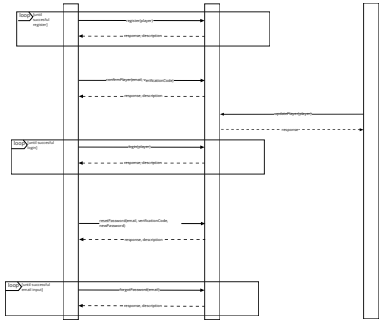


# Domain Models

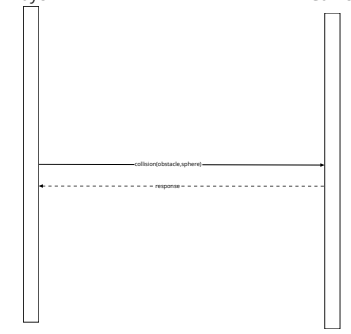


# System Sequence Diagrams

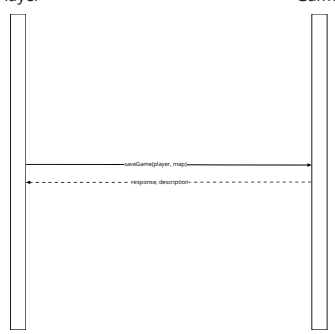
## Authorize



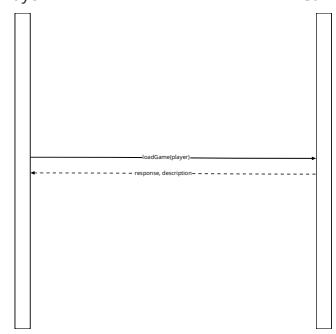

## Hit an Obstacle



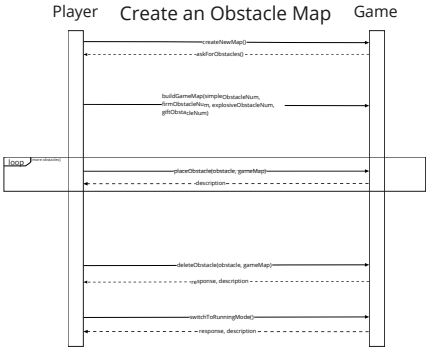

## Save the Game



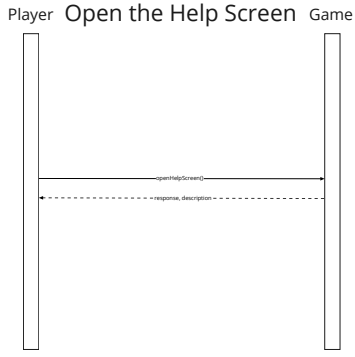

## Load the Game



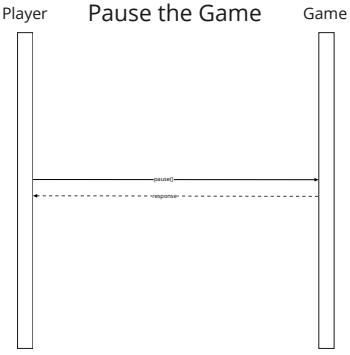

## Create an Obstacle Map




## Open the Help Screen

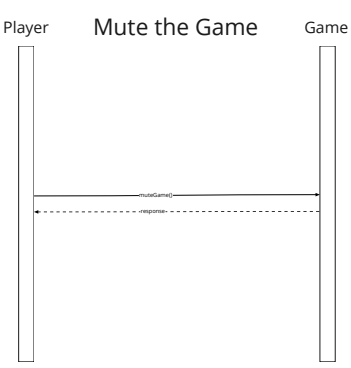



## Pause the Game

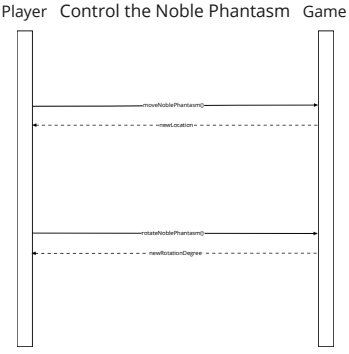



## Use a Spell

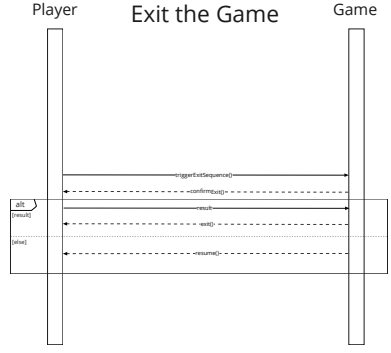

## Mute the Game



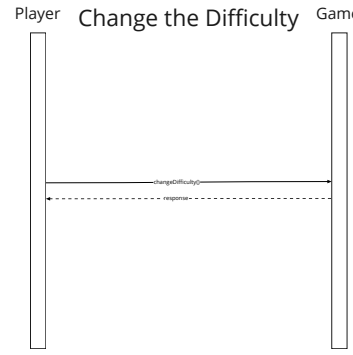

## Control the Noble Phantasm




## Exit the Game

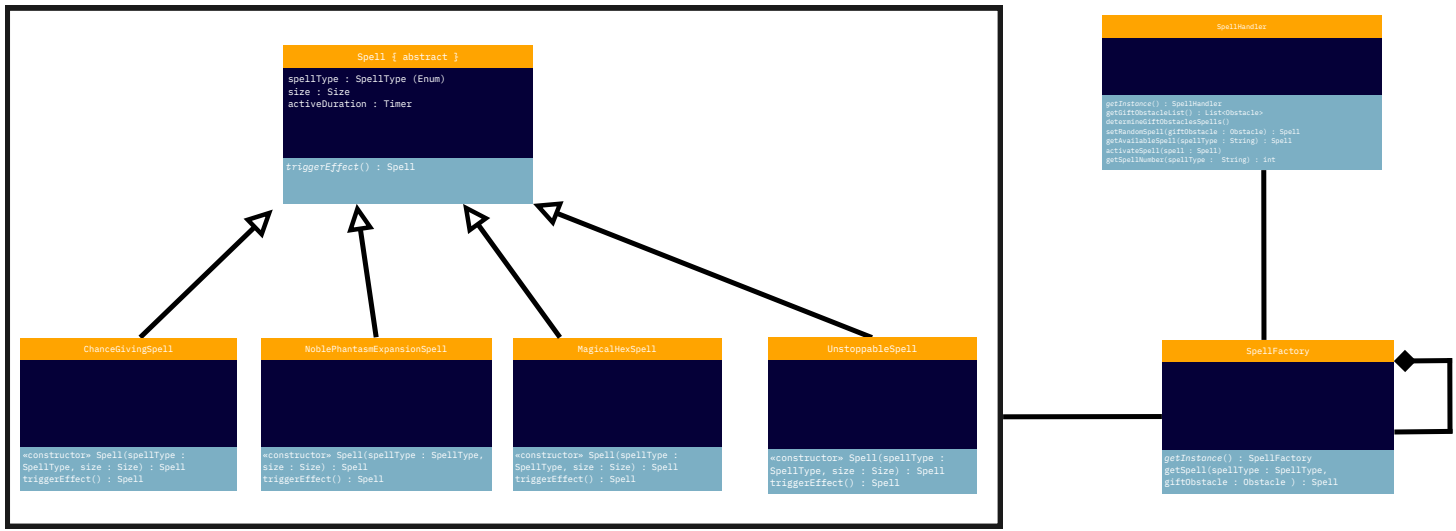



## Change the Difficulty

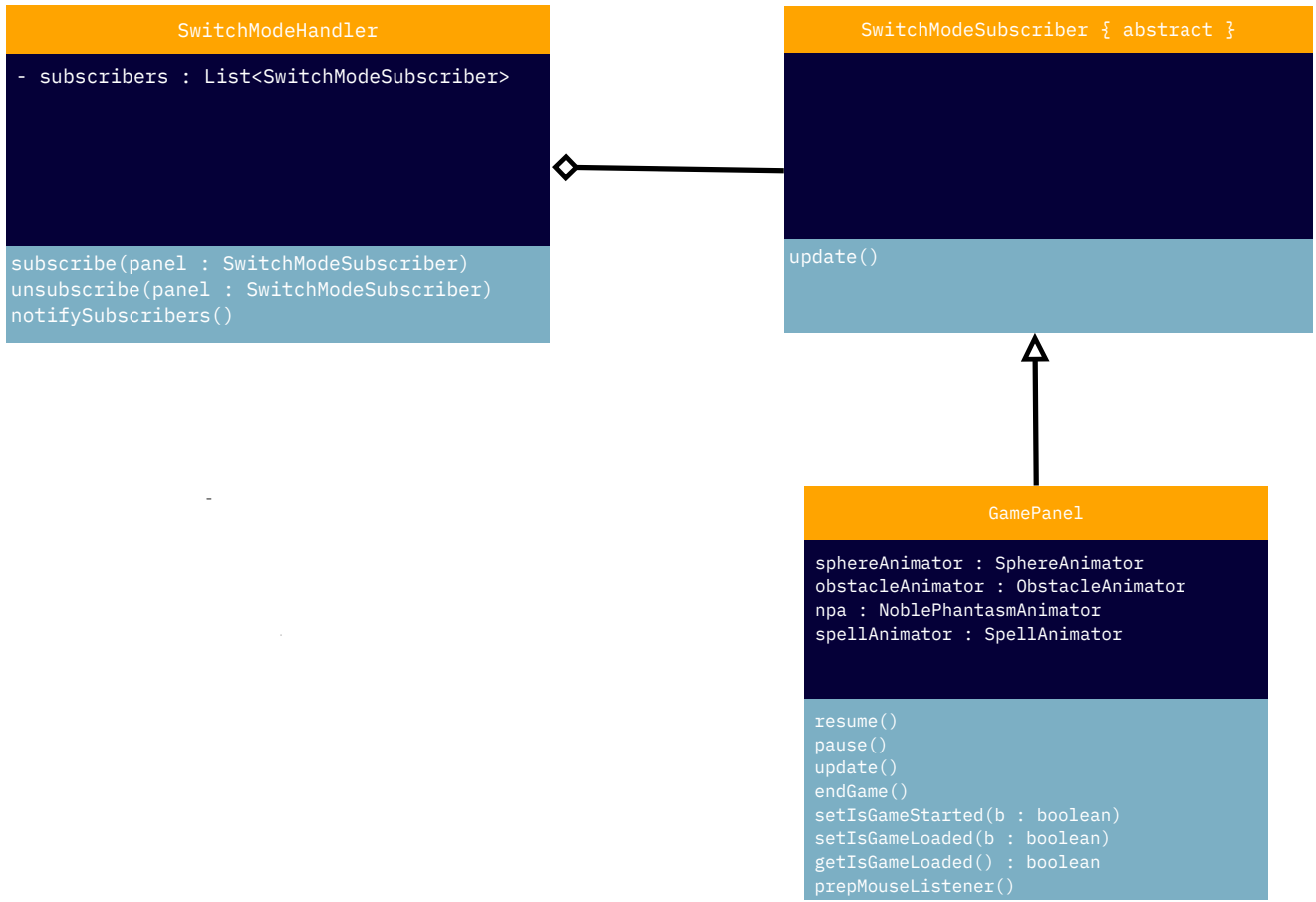


# Class Diagrams

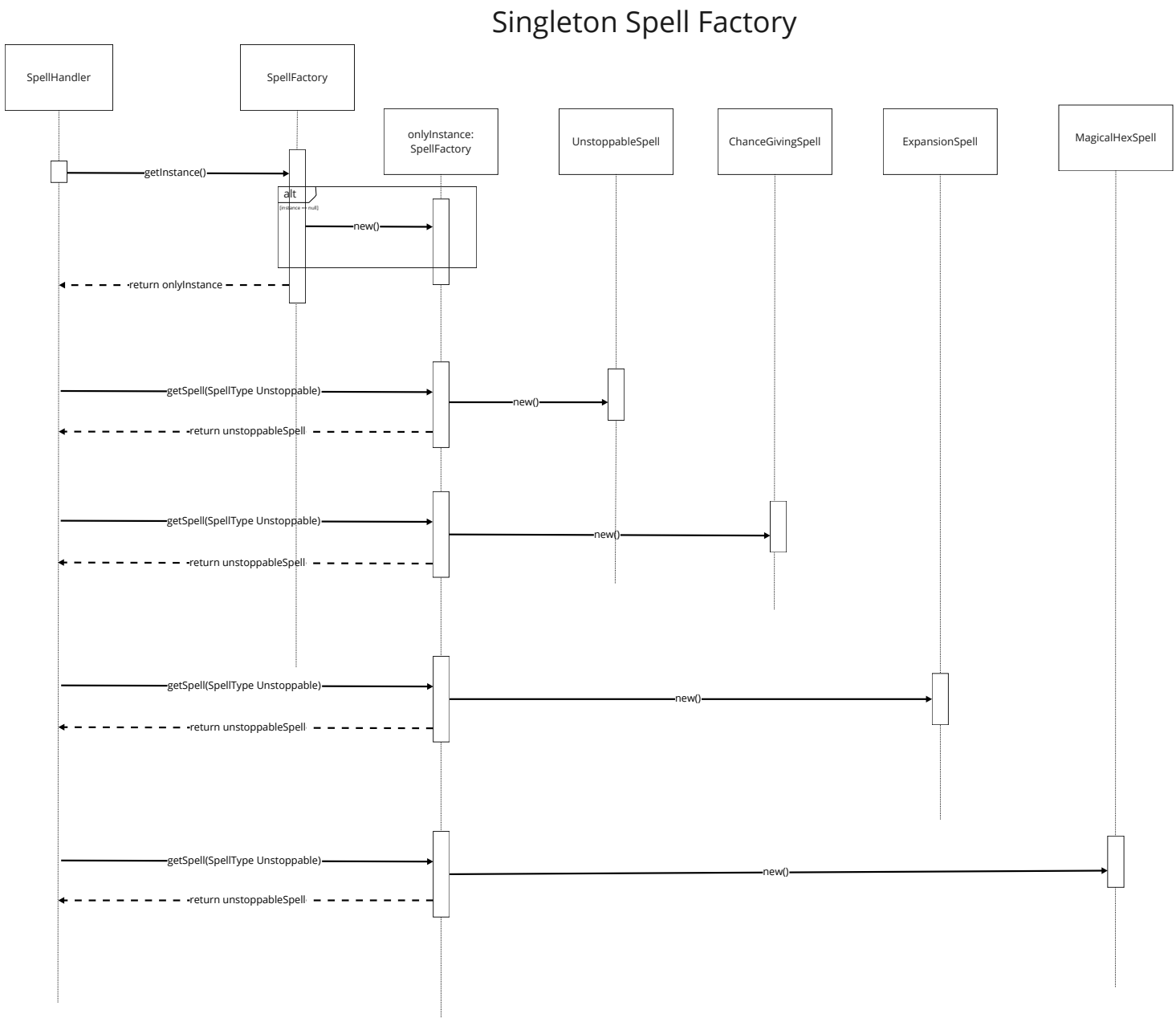
## Singleton Spell Factory



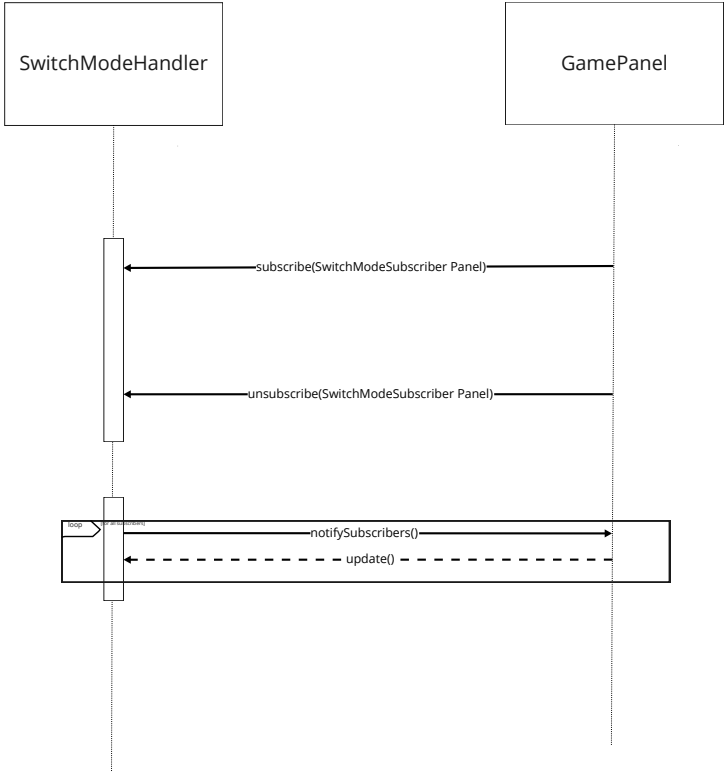
# Switch Mode Observer



Sequence Diagrams



# Observer


SwitchModeHandler
GamePanel
subscribe(SwitchModeSubscriber Panel)
unsubscribe(SwitchModeSubscriber Panel)
loop
notifySubscribers()
update()

# Need for Spear Use Cases

1. Authorize.
2. Hit an obstacle.
3. Save the game.
4. Load the game.
5. Open the help screen.
6. Pause the game.
7. Create an obstacle Map.
8. Use a spell.
9. Control the noble phantasm.
10. Exit the game.
11. Mute the game.
12. Change the difficulty.

**Use Case Narratives**

**Use Case UC1:** Authorize.
**Scope:** Login Screen.
**Level:** Subfunction.
**Primary Actor:** Player.
**Stakeholders and Interests:**

1. Player: Wants to create a mechanism to retrieve all personalized data about them. (Progress in the game, highscore etc.)

**Preconditions:**

1. Player must have an internet connection.
2. Player must be in the login screen.

**Success Guarantee:**

1. The player successfully logs in or, alternatively, signs up the game.

**Main Success Scenario:**

1. Player enters the username to the required area.
2. Player enters the password to the required area.
3. Player clicks to the "Login" button.
4. System checks if the username and password match.
5. System checks if the email is verified.
6. System directs the player to the Main Menu.

**Extensions:**

*****At any time**,** the player wants to sign up for the first time.

1. Player enters their email address.
2. Player enters a username.
3. Player enters a password.
4. System saves this information to the database and sends a verification email to the player.
5. Player verifies their email to complete its sign up process.

2a. If the player forgot their password, or wants to change it.

1. Player clicks the "Forgot my password" button.
2. System sends an email to the player.
3. Player selects a new password with the help of instructions sent by email.
4. System updates the password at the database.

4a. If the provided username is not found.

1. System informs the player that the provided username has not been authorized.

4b. Player enters a wrong password.

1. System notifies the player that the password is wrong and asks to try again.

5a. Player's mail is not verified.

1. System notifies the player that their mail is not verified.

**Special Requirements:**

1. Rectangular text fields should be large enough to take inputs.
2. Rectangular text fields should be visible through various backgrounds.

**Technology and Data Variations List:**

1. Username, email should be text field while password area should be secured field.
2. The input should be entered via keyboard.

**Frequency of Occurrence:** Whenever the player wants to login to the game.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC2:** Hit an Obstacle.

**Scope:** Running Mode.

**Level:** User Goal.

**Primary Actor:** Player.

**Stakeholders and Interests:**

1. Player: Wants to direct the moving sphere with the noble phantasm to hit the obstacles.

**Preconditions:**

1. Player has built the level or loaded a built one.
2. Player is in the running mode.
3. Player has one or more lives.
4. Player is logged in using their credentials.
5. Player must have an internet connection.

**Success Guarantee:**

1. The player successfully hits the obstacle with the sphere.

**Main Success Scenario:**

1. Player moves the noble phantasm according to the sphere's movement.
2. Player redirects the sphere by using the noble phantasm.
3. Sphere hits an obstacle.

**Extensions:**

1a. Player fails to hit the sphere with noble phantasm.

1. Player's chances are reduced by 1.

1a. Player loses the game if the number of chances reduces to 0.

3a. Sphere misses obstacles.

1. Sphere bounces back from the edges and comes back to the level of the noble phantasm.

3b. Player selects to use "Magical Hex" ability.

1. Player hits the obstacles using the ability instead of the sphere.

3c. Sphere hits "Firm Obstacle".

1. The obstacle's health number decreases by one.

1a. The obstacle is destroyed if its health is reduced to 0.

3d. Sphere hits “Explosive Obstacle”.

1. The obstacle’s remains drop downwards.
2. If the remains hit the noble phantasm, the player’s lives decrease by one.

3e. Sphere hits “Gift Obstacle”.

1. The obstacle drops a gift box downwards.
2. If the gift box touches the noble phantasm, the player gets a special ability.

2a. If the special ability is the “Chance Giving Ability”, the player's life increases by one.

2b. If the special ability is the “Noble Phantasm Expansion”, the noble phantasm’s size increases.

2c. If the special ability is the “Magical Hex”, the player can shoot using the cannons on both sides of the phantasm.

2d. If the special ability is the “Unstoppable Enchanted Sphere”, the sphere can penetrate the obstacles.

3. If the player already has unused abilities, the newly acquired ability is stored alongside the unused ones.

**Special Requirements:**

1. All the obstacles’ colors should be chosen such that they are visible between various background colors.
2. Obstacle’s health should be displayed on the obstacle either using a number or texture.
3. Gifted, normal and explosive obstacles should be distinguishable from each other.
4. Each special ability should appear at least once after every gift obstacle is destroyed.

**Technology and Data Variations List:**

1. Player should interact with the game via keyboard.

**Frequency of Occurrence:** Every time the player hits an obstacle.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC3:** Save the Game.

**Scope:** Building Mode/Running Mode.

**Level:** Subfunction.

**Primary Actor:** Player.

**Stakeholders and Interests:**

1. Player: Wants to save their progress and resume playing using that saved data.

**Preconditions:**

1. Player must logged in using their credentials.
2. Player must have an internet connection.
3. Player must have paused the game.

**Success Guarantee:**

1. Player save their progress or map without any exceptions.

**Main Success Scenario:**

1. Player clicks the “Save Game” button in the Overlay Panel.
2. Player’s current game state is saved in the database.

**Extensions:**

2a. Player already has a save entry in the database.

1. The current game state overrides the previous saved game entry.

**Special Requirements:**

1. Rectangular button should be visible through various backgrounds.

**Technology and Data Variations List:**

1. Player should save the game using keyboard and mouse.

**Frequency of Occurrence:** Whenever the player wants to save their game.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC4:** Load the Game.

**Scope:** Main Menu / Building Mode / Running Mode.

**Level:** Subfunction.

**Primary Actor:** Player.

**Stakeholders and Interests:**

1. Player: Wants to load a previously saved game without any exceptions.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in the Main Menu, Building Mode or Running Mode.
4. Player has paused the game.
5. Player has a previously saved game file.

**Success Guarantee:**

1. The player successfully loads the previously designed map and their progress.

**Main Success Scenario:**

1. Player clicks the “Load Game” button in the Main menu, Building Mode or Running Mode.
2. System retrieves the saved game file from the database.
3. System loads the previous game data from the database and starts the game with this data.

**Extensions:**

2a. Player has no previously saved game file.

1. Systems warns the user that there is no existing saved data.

2b. The desired game data is corrupted.

1. The player is alerted that the data is corrupted and cannot be loaded.
2. System deletes the corrupted data.

3. Player is sent back to the screen where they choose a save entry.

**Special Requirements:**

1. Rectangular button should be visible through various backgrounds.

**Technology and Data Variations List:**

1. Player should load the game using keyboard and mouse.

**Frequency of Occurrence:** Whenever the player wants to load their game.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC5:** Open the Help Screen.

**Scope:** Main menu.

**Level:** Subfunction.

**Primary Actor:** Player.

**Stakeholders and Interests:**

1. Player: Wants to get information about the game logic, movement and game progress interaction.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player is in the main menu.
3. Player must have an internet connection.

**Success Guarantee:**

1. Help screen pops up. The player gets information about the rules of the game and is informed about how to navigate through the interface.

**Main Success Scenario:**

1. Player clicks the “Help” button.
2. Help screen pops up.
3. Player exits the help screen.

**Extensions:**

**NO EXTENSION.**

**Special Requirements:**

1. Text font is appropriate for reading.
2. The text should be explanatory and clear.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.

**Frequency of Occurrence:** Whenever the player wants to get help.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC6:** Pause the Game.

**Scope:** Running mode

**Level:** Subfunction.

**Primary Actor:** Player.
**Stakeholders and Interests:**
1. Player: Wants to pause the ongoing game without any exceptions.

**Preconditions:**
1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in running mode.

**Success Guarantee:**
1. All of the animated obstacles, the sphere and the noble phantasm stop. Player's ability to control noble phantasm is no longer possible.

**Main Success Scenario:**
1. Player clicks the pause button on the game overlay while ingame.
2. The game is paused.

**Extensions:**
NO EXTENSION.

**Special Requirements:**
1. The player should be aware that the game can be paused and then resumed later.

**Technology and Data Variations List:**
1. Player should pause the game using the keyboard.

**Frequency of Occurrence:** Whenever the player wants to pause their game.
**Open Issues:** NO OPEN ISSUES.

**Use Case UC7:** Create an obstacle Map.
**Scope:** Building Mode.
**Level:** User Goal.
**Primary Actor:** Player.
**Stakeholders and Interests:**
1. Player: Wants to design a map which they want to save to play later or to play immediately.

**Preconditions:**
1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in building mode.

**Success Guarantee:**
1. Creates a new map in which the number, type and health of the bricks are set.

**Main Success Scenario:**
1. Player clicks the "New Game" button in the main menu.
2. An empty map, which contains different types of bricks for the player to choose, will be displayed.

3. Player designs the map by placing the bricks.
4. Player saves or plays the map.

**Extensions:**

3a. Player wants to remove a selected brick.

1. Player selects the brick and presses the right click.
2. System removes the brick from the screen.

3b. Player wants to put obstacles.

1. Player chooses the type of obstacle from the JComboBox.
2. Player clicks left twice.
3. System puts a new obstacle to the screen.

3a. If a new obstacle's location intersects with another, the new obstacle will not be placed.

**Special Requirements:**

1. All the bricks in the level should be visible.
2. Bricks' health should be set randomly.
3. Gifted, normal and explosive bricks should be distinguishable from each other.
4. Editing area for the player should be distinguishable.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.

**Frequency of Occurrence:** Whenever the player wants to design a map.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC8:** Use a Spell.

**Scope:** Running Mode.

**Level:** User Goal.

**Primary Actor:** Player.

**Stakeholders and Interests:**

Player: Uses a spell of their choosing.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in running mode.
4. Player has a spell.

**Success Guarantee:**

1. Player activates the spell.

**Main Success Scenario:**

1. Player clicks the icon or types the first letter of the spell.
2. Spell is activated.

**Extensions:**

1a. "Chance Giving Ability" is used immediately upon pickup.

**Special Requirements:**

1. Spells' icons should be visible and understandable.
2. Player should be able to understand whether they have any spells or not.
3. Player should be able to understand if there are any spells active.
4. Visualization of spells' effects should be distinguishable.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.
**Frequency of Occurrence:** Whenever the player wants to use a spell.
**Open Issues:** NO OPEN ISSUES.

**Use Case UC9:** Control the Noble Phantasm.
**Scope:** Running Mode.
**Level:** User Goal.
**Primary Actor:** Player.
**Stakeholders and Interests:**

1. Player: Wants to change the Noble Phantasm's location and degree.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in running mode.

**Success Guarantee:**

1. Player moves, rotates and speeds up the noble phantasm.

**Main Success Scenario:**

1. Player presses right or left arrows to move the phantasm horizontally.
2. Player presses the A and D keys to change the noble phantasm's rotation degree.
3. Player presses the down arrow to double the phantasm's speed.

**Extensions:**

1a. Player tries to control Noble Phantasm in pause mode.
    1. Phantasm's state is not changed.
1b. Player tries to move the phantasm out of the map.
    1. Phantasm stays at the edge of the map.

**Special Requirements:**

1. Noble Phantasm should be visible through various backgrounds.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.
**Frequency of Occurrence:** Whenever the player wants to move the phantasm.
**Open Issues:** NO OPEN ISSUES.

**Use Case UC10:** Exit The Game.
**Scope:** Main Menu
**Level:** Subfunction.
**Primary Actor:** Player.
**Stakeholders and Interests:**

1. Player: Wants to close the game without any exceptions.

**Preconditions:**

1. Player is in the main menu.

**Success Guarantee:**

1. Game shuts down.

**Main Success Scenario:**

1. Player clicks the "Exit" button.
2. A screen pops up asking "Are you sure?".
3. Player clicks "Yes".
4. Game closes.

**Extensions:**

2a. Player clicks "No"

2. Player returns to the previous screen.

**Special Requirements:**

1. Rectangular buttons should be visible through various backgrounds.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.
**Frequency of Occurrence:** Whenever the player wants to exit from the game.
**Open Issues:** NO OPEN ISSUES.

**Use Case UC11:** Mute the Game.
**Scope:** Running mode / Building mode.
**Level:** Subfunction.
**Primary Actor:** Player.
**Stakeholders and Interests:**

1. Player: Wants to mute the sound of the game.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in building mode or running mode.

**Success Guarantee:**

1. Sound of the game is muted.

**Main Success Scenario:**

1. Player clicks the "Mute Game" button on the screen.
2. The sound of the game is muted.

**Extensions:**

NO EXTENSION.

**Special Requirements:**

1. The player should understand that they can mute sound with a button.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.

**Frequency of Occurrence:** Whenever the player wants to mute the sound of their game.

**Open Issues:** NO OPEN ISSUES.

**Use Case UC12:** Change the difficulty.

**Scope:** Building Mode

**Level:** Subfunction.

**Primary Actor:** Player.

**Stakeholders and Interests:**

1. Player: Wants to change the difficulty of the game without any exceptions.

**Preconditions:**

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in building mode.

**Success Guarantee:**

1. Player alters the difficulty of the newly created game.

**Main Success Scenario:**

1. Player will choose the difficulty level on the popup menu on building mode.
2. The game difficulty will be changed.

**Extensions:**

NO EXTENSION.

**Special Requirements:**

1. The player should be aware of how the difficulty will be altered in the game.

**Technology and Data Variations List:** NO TECH or DATA VARIATION.

**Frequency of Occurrence:** Whenever the player creates a new map.

**Open Issues:** NO OPEN ISSUES.

**Operation Contracts**

**Operation:** register(*player* : Player*)*

**Cross References:** Authorize.

**Preconditions:**

1. Player has a valid email account.

**Postconditions:**

1. A Player instance *player* is created. *(instance creation)*
2. Email, username and password are associated with the *player* object. *(association formed)*

**Operation:** login(*player* : Player)
**Cross References**: Authorize.
**Preconditions:**

1. Username and password pair exists on the game database.

**Postconditions:**

1. *player* was associated with NeedForSpearGame. *(association formed)*
2. *player* was associated with GameMap. *(association formed)*
3. *player* was associated with Spell. *(association formed)*
4. *player* was associated with NoblePhantasm. *(association formed)*

**Operation:** forgotPassword(*email* : String)
**Cross References:** Authorize.
**Preconditions:**

1. *email* has an associated username and password *player* object on the game database.

**Postconditions:**

1. Verification code is sent to the player's email according to user input. *(attribute modification)*

**Operation:** confirmPlayer(email, verificationCode)
**Cross References:** Authorize.
**Preconditions:**

1. *email* has an associated username and password *player* object on the game database.

**Postconditions:**

1. *player*.isVerified is updated if the verificationCode is correct. *(attribute modification)*

**Operation:** resetPassword(email, verificationCode, newPassword)
**Cross References:** Authorize.
**Preconditions:**

1. *email* has an associated username and password *player* object on the game database.

**Postconditions:**

1. *player*.password is updated with a newPassword according to user input. *(attribute modification)*

**Operation:** collision(*obstacle* : Obstacle, *sphere* : Sphere)
**Cross References:** Hit an Obstacle.
**Preconditions:**

1. Game is in the running mode.
2. *player*.lives is 1 or more.

**Postconditions:**

1. *player*.score is updated accordingly if *obstacle*.health is 0 or below. *(attribute modification)*

**Operation:** loadGame(*player* : Player)
**Cross References:** Load the Game.
**Preconditions:**

1. Game is in the Main Menu or Running Mode.
2. There is a previously saved game file in the database.

**Postconditions:**

1. An instance of GameMap *map* was created. *(instance creation)*
2. *map.listOfObstacles is set to the data from the database. (attribute modification)*
3. *player*.score is set to score from the database. *(attribute modification)*
4. *player*.lives is set to lives from the database. *(attribute modification)*

**Operation:** createNewObstacle(*x* : double, *y* : double, *obstacleType :* String, *graphics :* Graphics, *obstacleAnimator :* ObstacleAnimator)
**Cross References:** Create an Obstacle Map.
**Preconditions:**

1. Game is in the build mode.
2. A GameMap instance was created previously.

**Postconditions:**

1. *obstacle* was associated with GameMap. *(association formed)*
2. *obstacle* is added to *map.listOfObstacles. (attribute modification)*

**Operation:** removeObstacle(*x* : int, *y* : int, *graphics :* Graphics, *obstacle:* Obstacle)
**Cross References:** Create an Obstacle Map.
**Preconditions:**

1. Game is in the build mode.

**Postconditions:**

1. *obstacle*'s association with the GameMap was broken. *(association broken)*
2. *obstacle is removed from map*.listOfObtacles. *(attribute modification)*

**Operation:** switchToRunningMode()
**Cross References:** Create an Obstacle Map.
**Preconditions:**

1. Game is in the build mode.
2. Obstacle conditions are met.

**Postconditions:**

1. Instances of NoblePhantasm and Sphere objects were created. *(instance creation)*
2. *NeedforSpearGame*.mode attribute was changed. *(attribute modification)*

**Operation:** activateSpell(*spell* : Spell)
**Cross References:** Use a Spell.
**Preconditions:**

1. Game is in the running mode.
2. Player should have at least one instance of spell.

**Postconditions:**

1. *spell* is associated with a Player, NoblePhantasm or Sphere object. *(association formed)*
2. *player.*listOfSpells is adjusted accordingly. *(attribute modification)*

**Operation**: moveNoblePhantasm()
**Cross References:** Control the Noble Phantasm.
**Preconditions:**

1. An instance of a NoblePhantasm *noblePhantasm* has been created.
2. An instance of Long *currentTime* is equalized to *currentTimeMillis* in System.
3. An instance of Long *lastUpdateTime* is equalized to *noblePhantasm.getLastUpdateTime.*
4. An instance of Long moveTimeDiff is equalized to *currentTime-lastUpdatedTime*.
5. System is in the running mode.

**Postconditions:**

1. *noblePhantasm*.isSpeeding is set according to the input. *(attribute modification)*
2. *noblePhantasm*.location is set according to the input. *(attribute modification)*

**Operation**: rotateNoblePhantasm()
**Cross References:** Control the Noble Phantasm.
**Preconditions:**

1. An instance of a NoblePhantasm *noblePhantasm* has been created.
2. An instance of Long *currentTime* is equalized to *currentTimeMillis* in System.
3. An instance of Long *lastUpdateTime* is equalized to *noblePhantasm.getLastUpdateTime.*
4. System is in the running mode.

**Postconditions:**

1. *noblePhantasm*.setRotationDegree is set to newAngle. *(attribute modification)*
2. noblePhantasm.setUpdateTime is set to currentTime. *(attribute modification)*

Operation: muteGame()
Cross References: Mute the Game.
Preconditions:

1. An instance of a ScheduledExecutorService *is created.*
2. *An instance of music and background sound thread is created.*

Postconditions:

1. *The instance of the music and sound effect threads are destroyed. (instance destroyed)*

Operation: changeDifficulty(*difficulty* : String)
Cross References: Change the Difficulty.
Preconditions:

1. Game is in Building mode.

Postconditions:

1. NeedForSpearGame.difficulty is set to the *difficulty. (attribute modification)*
2. GamePanel.delay attribute is changed according to the *difficulty. (attribute modification)*