# Design Overview for Descend Below (Lite)

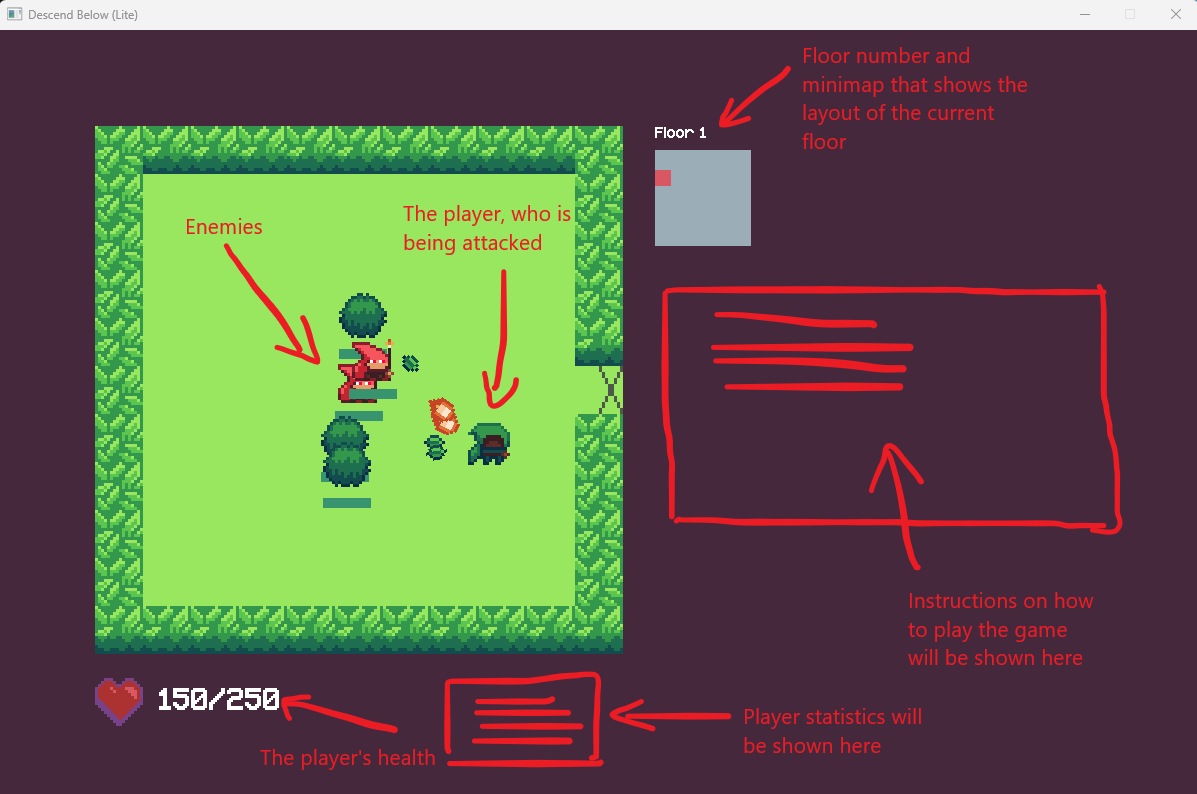
Name: Ta Quang Tung

Student ID: 104222196

# Summary of Program

Descend Below (Lite) is a roguelike video game where the player battle enemies to progress through endless floors. Each floor is made up of rooms, which contain enemies and exits that lead to other rooms. One room in each floor contains a staircase that can be clicked on to enter a new floor.

The player attack enemies by left clicking. Enemies attack the player by firing projectiles at them. The game ends when the player’s health reaches 0, after which the game can be reset.



*Figure 1: Sketch of final output (more details will be added later)*

# Required Roles

Table : GameObject abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| \_position | protected field, Point2D | The position of the game object. |
| \_width | protected field, double | The width of the game object. |
| \_height | protected field, double | The height of the game object. |
| \_sprite | protected field, Bitmap | The sprite (image) of the game object. |
| \_zIndex | protected field, int | A number that controls how the game object will be drawn in relation to other objects. Objects with higher z-indices are drawn on top. |
| GameObject | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, int zIndex = 1) | Used to create a new game object. |
| Draw | public virtual method, parameter (DrawingOptions options), returns void | Draws the game object to the screen, using the provided DrawingOptions. |
| Update | public abstract method, parameter (uint fps), returns void | Updates the game object. |

Table 2: Collider class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| \_gameObject | private field, GameObject | The game object associated with the collider. |
| \_baseColliderBox | private field, Quad | A rectangle representing the collider whose center is rooted at (0, 0). The Quad type is chosen over Rectangle because the collider can be rotated. |
| Collider | public constructor, parameters (GameObject gameObject, double rotation) | Used to create a new collider object. |
| IsCollidingWith | public method, parameter (Collider c), returns bool | Determines if the collider is colliding with another collider. |
| GetColliderBox | private method, no paremeters, returns Quad | Returns a new Quad after moving the base collider box to the position of the game object. |
| GameObject | public readonly property, returns GameObject | Used to retrieve the game object associated with the collider. |

Table 3: StaticObject abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| … StaticObject inherits from GameObject | | |
| StaticObject | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, int zIndex = 1) | Used to create a new static game object. |
| Update | public override method, parameter (uint fps), returns void | Updates the static game object. |

Table 4: DynamicObject abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| … DynamicObject inherits from GameObject | | |
| \_velocity | protected field, Vector2D | The velocity of the dynamic object. |
| DynamicObject | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, Vector2D velocity, int zIndex = 1) | Used to create a new dynamic game object. |
| Draw | public override method, parameter (DrawingOptions options), returns void | Draws the dynamic object, flipping the sprite across the Y axis depending on its FacingDirection. |
| Update | public override method, parameter (uint fps), returns void | Updates the dynamic game object, moving it by its velocity. |
| MoveTo | public method, parameter (Point2D point), returns void | Moves the object to the specified position. |
| MoveBy | public method, parameter (Vector2D displacement), returns void | Moves the object along the specified vector. |
| GetFacingDirection | protected virtual method, no parameters, returns FacingDirection | Determines the FacingDirection of the dynamic object, which can be used for drawing. |

Table 5: Projectile class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| … Projectile inherits from DynamicObject and implements the ICollidable and IDestroyable interfaces | | |
| \_rotation | private field, double | The projectile’s rotation measured counter-clockwise from the vector (1, 0). Used to draw the projectile and set up its collider. |
| \_collider | private field, Collider | The Collider object associated with the projectile. |
| \_canDestroy | private field, bool | Whether the projectile can be destroyed or not. |
| \_projectileType | private field, ProjectileType | The type of the projectile. Can be either Friendly or Hostile. |
| \_damage | private field, int | The projectile’s damage. |
| Projectile | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, Vector2D initialVelocity, double targetSpeed, ProjectileType type, int damage) | Used to create a new projectile. |
| Draw | public override method, parameter (DrawingOptions options), returns void | Draws the projectile, rotating it by \_rotation. |
| Collide | public virtual method, parameter (Collider c), returns void | Defines the projectile’s behavior on colliding with an object. |
| Collider | public readonly property, returns Collider | Used to retrieve the projectile’s collider. |
| CanDestroy | public readonly property, returns bool | Used to determine if the projectile can be destroyed. |

Table 6: ICollidable interface details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Collider | readonly property, returns Collider | Used to retrieve the collider (a hitbox) associated with a game object. |
| Collide | method, parameter (Collider collider), returns void | A method that defines the object’s behavior upon collision with another object. |

Table 7: IDestroyable interface details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| CanDestroy | readonly property, returns bool | Used to determine whether an object can be destroyed or not. Destroying means removing all references to an object. |

Table 8: Character abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Character inherits from DynamicObject and ICollidable | | |
| \_health | protected field, int | The character’s current health. |
| \_maxHealth | protected field, int | The character’s maximum health. |
| \_collider | private field, Collider | The collider associated with the character. |
| Character | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, Vector2D initialVelocity, int maxHealth, int zIndex = 1) | Used to create a new character. |
| Collider | public readonly property, returns Collider | Used to retrieve the character’s collider. |
| Collide | public virtual method, parameters (Collider c), returns void | Defines the character’s behavior on colliding with an object. |
| Damage | public virtual method, parameters (int amount), returns void | Damages the character by a particular amount. |
| Heal | public virtual method, parameters (int amount), returns void | Heals the character by a particular amount. |
| IsDead | public method, no parameters, returns bool | Used to determine if the character is dead. |

Table 9: Player class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Player inherits from Character | | |
| \_weapon | private field, Weapon | The player’s equipped weapon. |
| Player | public constructor, parameters (Point2D position, Vector2D initialVelocity, int maxHealth) | Used to create a new player. |
| Halt | public method, returns void | Sets the player’s velocity to 0. |
| MoveAlong | public method, parameters (Vector2D direction) | Sets the player’s velocity to move along the specified direction. |
| Attack | public method, parameters (Point2D target) | Attacks a target at the specified location. |

Table 10: Enemy abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Enemy inherits from Character and IDestroyable | | |
| Enemy | public constructor, parameters (Point2D position, double width, double height, Bitmap sprite, Vector2D initialVelocity, int maxHealth) | Used to create a new enemy. |
| Attack | public abstract method, parameters (Player p) | Attacks the player specified by p. Must be implemented by derived classes. |
| Move | public virtual method, parameters (Player p) | Moves towards the player specified by p. |
| CanDestroy | public readonly property, returns bool | Used to determine if the enemy object can be destroyed. |

Table 11: Wall class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Wall inherits from StaticObject and ICollidable | | |
| \_collider | private field, Collider | The collider object associated with the wall. |
| Wall | public constructor, parameters Point2D position, double width, double height, Bitmap sprite | Used to create a new wall. |
| Collider | public readonly property, returns Collider | Used to retrieve \_collider. |
| Collide | public method, parameter Collider c | Defines the wall’s behavior on collision with another object. |

Table 12: Exit class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Exit inherits from StaticObject and ICollidable | | |
| \_collider | private field, Collider | The collider object associated with the exit. |
| \_sourceRoom, \_destinationRoom | private fields, Room | The source and destination rooms associated with the exit. |
| \_direction | private field, Direction | The direction of the exit. |
| Exit | public constructor, parameters Point2D position, double width, double height, Bitmap sprite, Direction direction, Room sourceRoom, Room destinationRoom | Used to create a new exit. |
| Collider | public readonly property, returns Collider | Used to retrieve \_collider. |
| Collide | public method, parameter Collider c | Defines the exit’s behavior on collision with another object. |

Table 13: Interactable abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Interactable inherits from StaticObject | | |
| \_range | private field, double | The range within which the object can be interacted with. |
| Interactable | public constructor, parameters Point2D position, double width, double height, Bitmap sprite, double range | Used to create a new interactable object. |
| IsNearPlayer | public method, parameter Player p, returns bool | Determines whether the specified player is within range of the interactable. |
| IsHoveredOn | public method, parameter Point2D mousePosition, returns bool | Determines whether the mouse is on top of the interactable. |
| HandleInteraction | public abstract method, returns void | Defines the behavior of the interactable when it is clicked on. Must be implemented by derived classes. |

Table 14: Staircase class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Staircase inherits from Interactable | | |
| Staircase | public constructor, parameter Point2D position | Used to create a new staircase object. |
| HandleInteraction | public override method, returns void | Defines the behavior when the staircase is clicked on. |

Table 15: Weapon abstract class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| \_damage | private field, int | The weapon’s damage |
| \_attackCooldown | private field, double | The weapon’s attack cooldown in seconds. |
| Weapon | public constructor, parameters int damage, double attackCooldown | Used to create a new weapon object. |
| ReadyForAttack | protected method, returns bool | Determines if the weapon is off cooldown for an attack. |
| IncurCooldown | protected method, returns void | Puts the weapon on cooldown after an attack. |
| Attack | public abstract method, parameters Point2D startPosition, Point2D target | Attacks the target at the specified location. Must be implemented by derived classes. |

Table 16: Bow class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Bow inherits from Weapon | | |
| Bow | public constructor, parameters int damage, double attackCooldown | Used to create a new bow object. |
| Attack | public override method, parameters Point2D startPosition, Point2D target | Defines the bow’s attack behavior. |

Table 17: Room class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| \_gameObjects | private field, List<GameObject> | The list of game objects in the room. |
| Room | private constructor | Used to create a new Room object. Can only be used inside the Room class. |
| CreateRoom | public static method, parameters bool hasNorthExit, bool hasEastExit, bool hasSouthExit, bool hasWestExit, returns Room | Creates a normal room. Use this method to create a new room outside the Room class. |
| CreateEndRoom | public static method, parameters bool hasNorthExit, bool hasEastExit, bool hasSouthExit, bool hasWestExit, returns Room | Creates a room with a staircase that can be clicked on to enter a new floor. Use this method to create a new room outside the Room class. |
| IsClear | public method, return bool | Determines if a room is cleared of all enemies. |
| GameObjects | public readonly property, returns List<GameObject> | Used to retrieve the list of game objects in the room. |
| AddExit | public method, parameters Direction direction, Room destination | Adds an exit in the specified direction that leads to the specified room. |

Table 18: Floor class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| \_rooms | private field, Array of Rooms | A two-dimensional array of rooms in the floor. |
| \_startRoom | private field, Room | The starting room of the floor. |
| Floor | private constructor | Can only be used inside the class to create a new floor. |
| CreateFloor | public static method, returns Floor | Used to create a new floor outside the Floor class. |
| DrawMinimap | public method, parameters double x, double y, Room currentRoom | Draws the minimap showing the floor layout. |
| StartRoom | public readonly property, returns Room | Used to retrieve the starting room. |

Table 19: Game class details

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| CurrentGame | public static field, Game? | The current and only running game instance. |
| \_window | private field, Window | The game window. |
| \_state | private field, GameState | The state of the game. |
| \_floor | private field, Floor | The current floor object. |
| \_floorCounter | private field, int | The current floor number. |
| \_currentRoom | private field, Room | The current room the player is in. |
| \_objectsOnScreen | private field, List<GameObject> | The list of game objects currently on the screen. |
| \_player | private field, Player | The active player. |
| Game | private constructor | Used to create a new Game instance from inside the class. |
| CreateGame | public static method, returns Game | Creates and returns a new Game instance if there is none, otherwise returns CurrentGame. |
| Run | public method, returns void | Runs the game loop, which consists of: handling inputs, updating the game logic, handling collisions, and drawing objects. |
| CleanUp | public method, returns void | Cleans up the resources loaded for the game. |
| LoadResources | private method, returns void | Loads the game resources. |
| HandleInputs | private method, returns void | Handles the user’s inputs. |
| Update | private method, returns void | Updates the game logic. |
| HandleCollisions | private method, returns void | Handles collisions between game objects. |
| Draw | private method, returns void | Draws game objects onto the screen. |
| AddGameObjectOnScreen | public method, parameter GameObject gameObject, returns void | Used to add a game object onto the screen (e.g., a projectile). |
| EnterRoom | public method, parameters Room room, Direction enterDirection, returns void | Enters the specified room from the specified direction. |
| CurrentPlayer | public readonly property, returns Player | Used to retrieve the active player. |
| EnterNewFloor | public method, returns void | Creates and enters a new floor, incrementing the floor number. |
| ResetGame | private method, returns void | Resets the game after the player has died. |

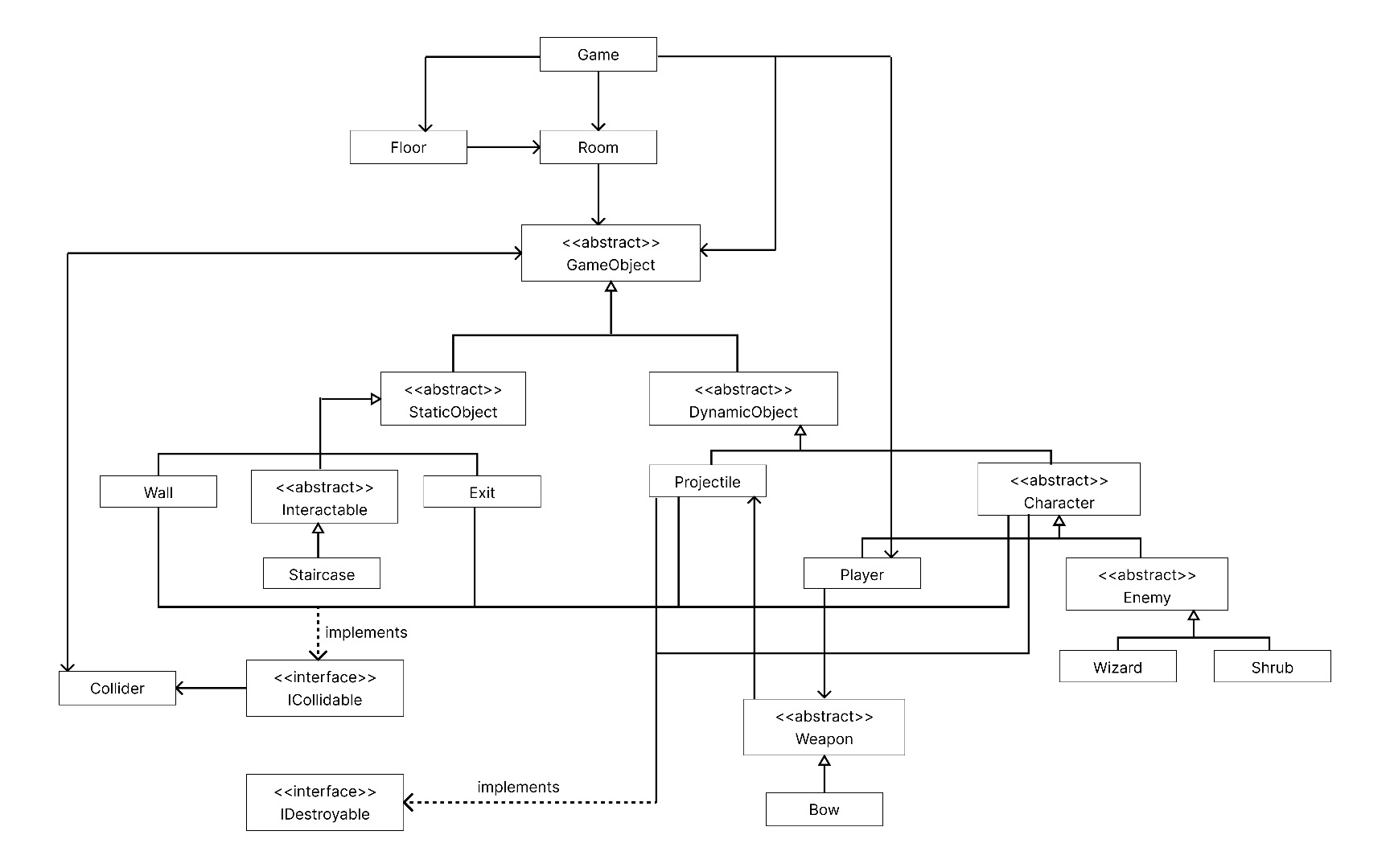
Table 20: Direction enumeration details

|  |  |
| --- | --- |
| Value | Notes |
| North | Cardinal directions of exit gates. |
| East |
| South |
| West |

Table 21: GameState enumeration details

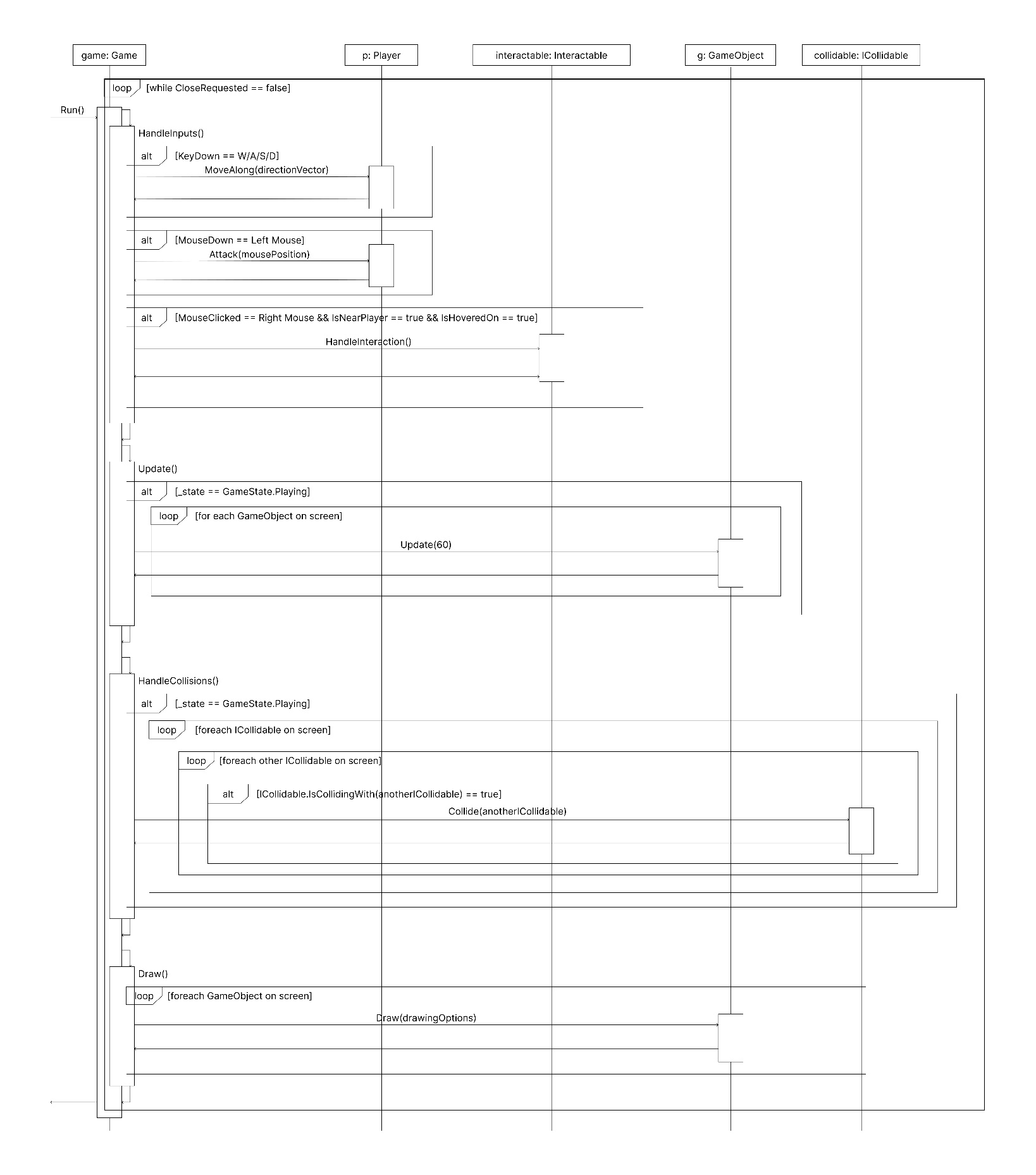
|  |  |
| --- | --- |
| Value | Notes |
| Playing | The player can press ESC to pause/unpause. |
| Paused |
| Lost | The game state is Lost when the player’s health reaches 0. |

# Class Diagram



*The program’s class diagram. Details of the classes and interfaces are shown in the previous section.*

# Sequence Diagram



*This sequence diagram shows how the game loop works. The Run method of the Game instance is called to initiate the game loop. The loop terminates when the user closes the game window.*