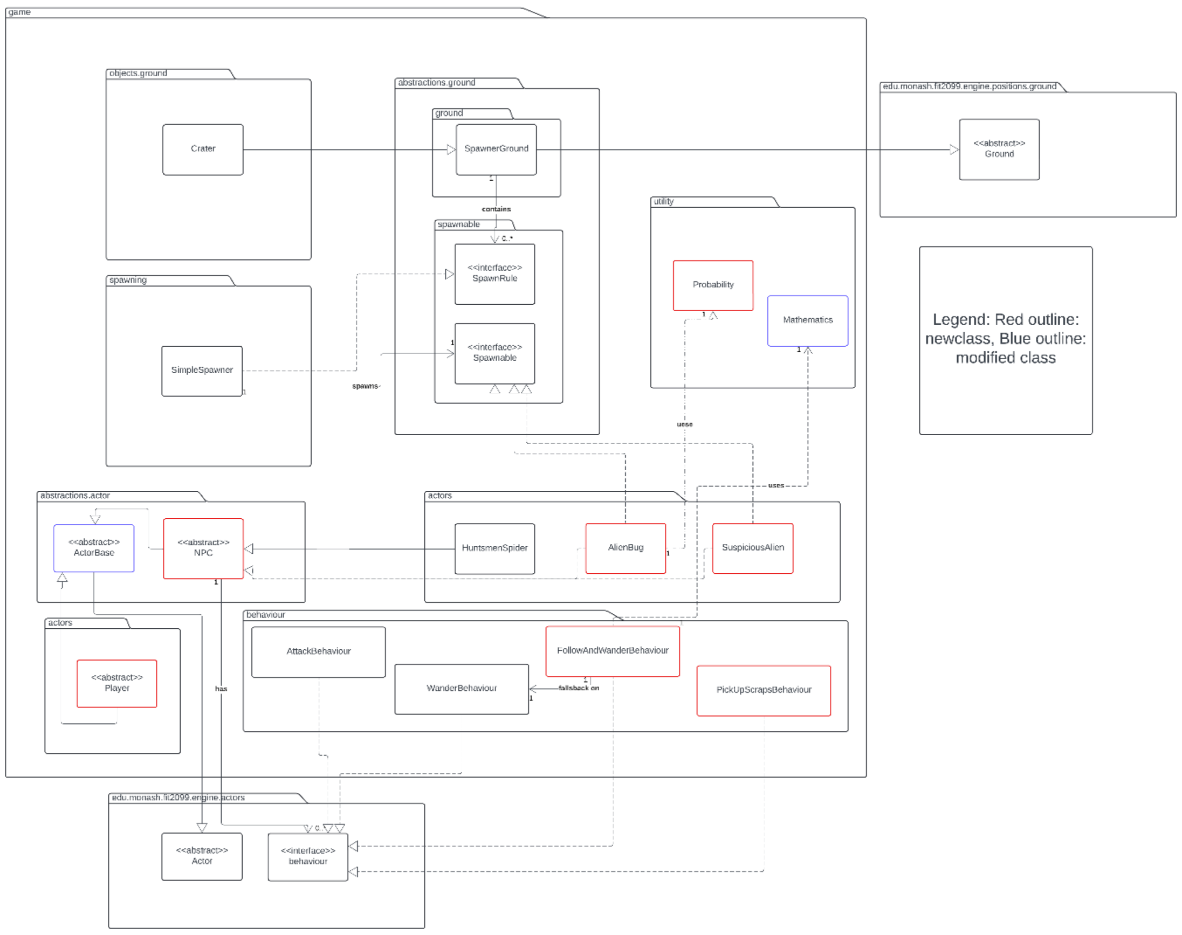
Assignment 2 Req2



Design Goal
The overall design goal was to keep the current code there and build upon it however, refactors could occur if there was enough of a push for them. The first thing that was major overhaul in the design is the transforming an object, we created a transformable interface, this mean that all objects that implemented transformable had default method which checked whether the conditions were correct to transform. This is good as other classes do not have to implement transformable, in the previous design the Tree class control transforming, but that meant that all classes that derived from that had to implement methods that would be useless to them if they do not transform. This was not in this requirement but is something to bring up.

Design Decisions
REQ2 introduces the alien bug and Suspicious Astronaut entity, implemented as Spawnable for crater being able to spawn it via SpawnRules and Spawner. Each actor class defines its behavior through separate Behavior classes (FollowAndWanderBehaviour, PickUpScrapsBehaviour, etc). By encapsulating behavior into separate classes, the design promotes modularity and reusability, allowing for easier maintenance and extension of the codebase.

Alien bug features two behaviors: PickUpScrapsBehavior and FollowAndWanderBehavior, both implementing the Behavior interface. PickUpScrapsBehavior simply checks for items at the actor's location, offering simplicity and reliability. FollowAndWanderBehavior, on the other hand, checks for nearby interns to follow. It utilizes a composition of WanderBehavior for flexibility and code reuse, when there is no Actors to follow.

Items dropping upon actor death is handled in the unconscious method of a new class. This is because Actor cannot be modified where it would make most logically sense, therefore we've made a new class called ActorBase, and we've chosen to override the unconious. We also considered adding the logic to the attackAction, however, this violated SRP, and therefore moved this to the ActorBase class.

Suspicious Astronaut entity on the other hand reused the attackBehaviour and added the capability to attack the intern. also used Integer.MAX_VALUE to deal ultimate dmg.

Final Design

The final design adheres to SOILD and DRY principles, with all repeated code occurring in a base class or a composition class. Each individual class is responsible for a single task, such as attackaction, dealing with the attack logic and handing off death logic to the actor class. Moreover, the code is extendable easily with classes being able to inherit or to be extended for further functionality and closed for modifying. All classes also maintain LSP, as all subclasses implement all expect functionality expect from the base classes.