

Requirement 1: The moon's (hostile) fauna II



Design Decision

Alternative Design

An alternative design to implementing the creature classes and their respective spawning attributes may see the NPC classes have their 'spawn chance' as an attribute of the class. However, the use of magic number (spawn chance) leads to obscurity in design intention and makes it difficult for future extension, thus this design approach was not chosen.

```
public class AlienBug extends NPC implements Spawnable {  
    private final double spawnChance = 0.1;  
    public AlienBug() { super ( "Alien Bug", 'a', 2 ); }  
}
```

Analysis of Alternative Design

1. Single Responsibility Principle

- The spawn logic should be delegated to a designated spawn interface that handles the responsibility as opposed to the creature class.

Final Design

The chosen design better adheres to SOLID Principles while maintaining a robust codebase and ensuring multiple inheritance for easier future extension.

1. DRY

- As AlienBug and SuspiciousAlien extend NPC, the codebase is more robust through extending these identities maintaining readability. This allows the creatures share the same base functionality but with different behaviours.

2. Open – Closed

- Through the abstraction of the spawning logic, the addition of 'spawnable' creatures demonstrates extending the existing implementation, without further modification.

3. Liskov – Substitution

- As the creature classes implement Spawnable, the upcasting of NPC objects when passing to a Crater object allows for different creatures to be spawned by Craters during the game. This allows for flexibility in future extensions as there may be various creatures that can spawn from a given ground object.

4. Single – Responsibility

- Utilising an interface to delegate spawning logic; as a result creature instances are not responsible their spawning implementation. This ensures the code is more scalable and extensible in future iterations.