**Proxy Design Pattern**

Problem:

We need to create 55 Aliens in Space Invaders by a combination of 11 Squids, 22 Crabs, and 22 Octopuses. Additionally, a similar issue with the shields was reusing the same sprite multiple times.

Solution:

My approach is to avoid creating multiple sprites with excessive data. Instead, we can reduce unnecessary overhead by using a proxy pattern. The proxy pattern acts as a reference to the real sprite but holds only a small amount of information.

Pattern Description:

Diagram, schematic

Description automatically generated

The above UML diagram is an example for Proxy pattern that points to the Original Sprite.

The Proxy pattern is a design pattern that provides a substitute or placeholder for another object. It allows us to provide a level of indirection between the client code and the real object. The Proxy can control access to the real object, limit access to its resources, and provide a simplified interface to the client code. It's useful for improving performance, protecting sensitive information, and simplifying the client interface.

Key Object-Oriented mechanics:

Implementing the proxy pattern is a straightforward process since it closely resembles the real subject, with the only difference being that it holds less data. Consider an abstract class with abstract methods, and a subclass called Subject that extends that abstract class and contains additional information. If the client requires the creation of similar objects with slight variations in information, a proxy subject class can be used to point to the original subject class with only the necessary information. This is possible because the proxy has access to the original subject.

The Proxy pattern has three main components: the Base Subject, which is an interface for the abstract methods; the Proxy Subject, a concrete class that extends the Base Subject and holds a reference to the Real Subject; and the Real Subject, another concrete class that holds actual data.

Uses:

The Proxy pattern was used in Space Invaders to create similar sprite objects with different positions using the SpriteProxy class. Rather than creating multiple sprite objects, a sprite proxy was used to link to the original sprite, reducing the overhead of the original sprite class. The original sprite pointed to Azul Sprite, Color, and Rect, all of which required significant amounts of memory.

In contrast, SpriteProxy only contained position information and a pointer to the real sprite, resulting in less memory usage. Additionally, a SpriteBoxProxy was created to do the same thing but with less memory usage.