GSIP Pre-Proposal

Sad Pumpkin Combat Engine

Brief

The Sad Pumpkin Games Combat Engine is a game-design agnostic, generic combat engine for use in developing turn-based games across a wide variety of genres. The engine is developed in such a way that it is entirely game-design agnostic, meaning that it will execute turn-based combat regardless of the game built atop it. So long as each entity in the combat implements the basic 'actor' definition, combat will execute using a flexible initiative order system and prompt each entity to take their turn. Each 'actor' must implement certain low-level behaviors in order to make use of the Combat Engine, which include supplying available actions, initiative modifier, and basic status information. This low-level, adaptable engine guarantees that games of any genre can be created based off the same combat engine.

Technology

The Sad Pumpkin Combat Engine is built in .NET and heavily uses industry-standard design patterns.

Signal Pattern

Since the Sad Pumpkin Combat Engine is a game-agnostic utility object, it heavily uses the Signal/Slots pattern in order to negate the need for direct dependencies as much as possible. The Signal Pattern also allows consumers of Combat Engine to better abstract their implementation and to avoid threading issues by using a thread-safe signal implementation.

Null Object Pattern

The Sad Pumpkin Combat Engine makes heavy use of interfaces and has few concrete implementations as a part of its code. To allow for better testing of the Combat Engine and to avoid unnecessary conditionals, all interfaces in the Combat Engine follow the Null Object Pattern.

Adapter Pattern

As mentioned in the brief, the Sad Pumpkin Combat Engine heavily uses interfaces to define contracts between the game-specific code and the engine itself. While not directly implemented in the combat engine, the Adapter pattern is heavily used by consumers of the engine in order to define entities which implement multiple interfaces: such as 'actor', 'initiative provider', and 'targetable'.

Multithreading

The Sad Pumpkin Combat Engine uses multithreading in order to avoid blocking on the rendering thread. All the runtime components of the engine are by default performed on a worker thread and send signals back to the caller thread in order to communicate state changes and prompt for an actor's action selection.