Retro Hell

September 2018

Bullet Hell / Roguelike / Procedural / Arcade / Survival / Dungeon Crawler

(In need of better title)

Main Mechanics

Procedural Map Generation

The inventory takes shape of the Toolbelt of the adventurous player.
Since the pacing of the game is centered around fast and quick, the UI and inventory will be designed to convey quick, ease-of-use and simplicity to the player. Hence the player has about 3-5 inventory slots, which are easily scrolled through with the Q and E buttons.

Crafting System & Inventory

The inventory takes shape of the Toolbelt of the adventurous player.
Since the pacing of the game is centered around fast and quick, the UI and inventory will be designed to convey quick, ease-of-use and simplicity to the player. Hence the player has about 3-5 inventory slots, which are easily scrolled through with the Q and E buttons.

The Crafting system works entirely by easy keypresses. Like the inventory, it is designed for fast paced action. By combining two items you can swiftly craft a new item, which gets placed in the first slot (from left to right) that's empty in the player's inventory.

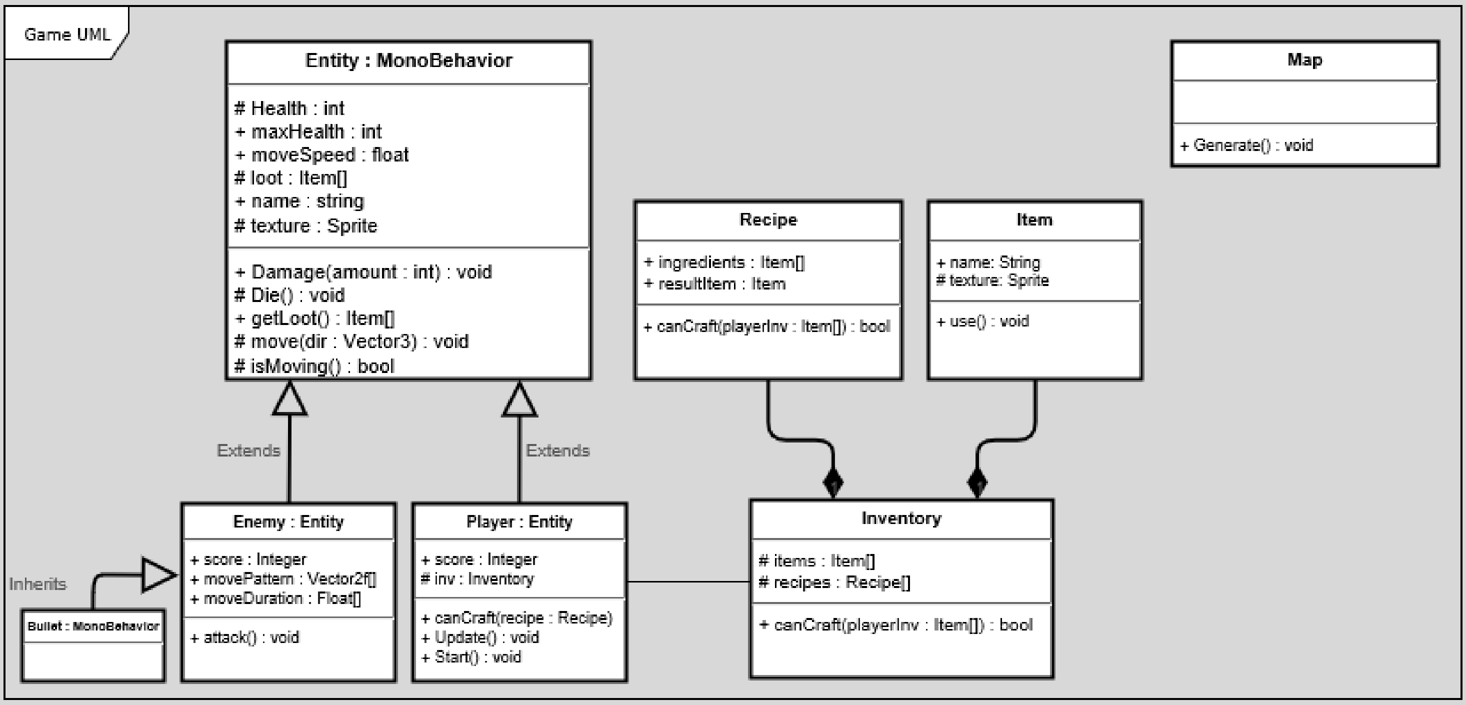
Bullets, Monsters & Items

The 3D world is littered with also procedurally generated 2D tiled dungeons. Once the player enters said dungeon to gain some new items / EXP, monsters will spawn / bullet hell sequence will begin.

The combat works by mouse or keyboard (Haven't decided yet).

Depending on the item, your hitting range, power, speed of attack and knockback will be increased/decreased. But most items will be focused on melee combat, and ranged weapons will be less common.

Planning of Design Strategy



For this project I will use the Unity engine, and make some base classes (such as Map, Entity, Inventory, etc.) as the foundation of the game.

Eventually the map will place random item and entity spawns and generate dungeons / structures for exploration.

The important part of the technical design is the Inventory system. And I think the *Façade* Design pattern fits this the best.

The inventory class will resemble a singleton, however the system will have all the necessary functions run in the background, therefore making it a façade design pattern. The user will be able to non-directly communicate with the Inventory slots, items and arrangements by using the public functions within the Inventory class handler.

For entities, the player and monster class will inherit from the abstract Entity class, which handles the basic functions of an entity within the world such as Health management, rendering and basic movement.