Combat Design Document

Designer:	Yexiang Zhou
Document Date:	11/11/2024

Table of Contents

Quick Summary

The purpose of this design sample and demo is to build a basic action combat framework, providing an easy-to-learn action and friendly game experience for casual players.
Specifically, it has:

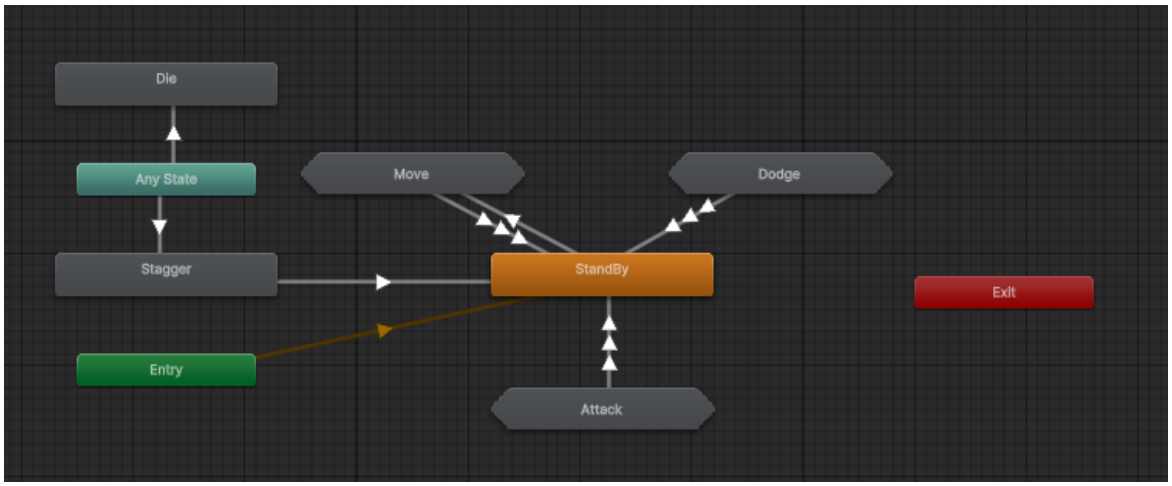
- Low complexity (width) and probability space, which means fewer rules and elements.
- High active action frequency and low passive action frequency , to let the player take control of combat pacing.
- Resource design focus on player behavior.
- AI and game mode design focus on controllable combat intensity.

Core Mechanics

States and transitions

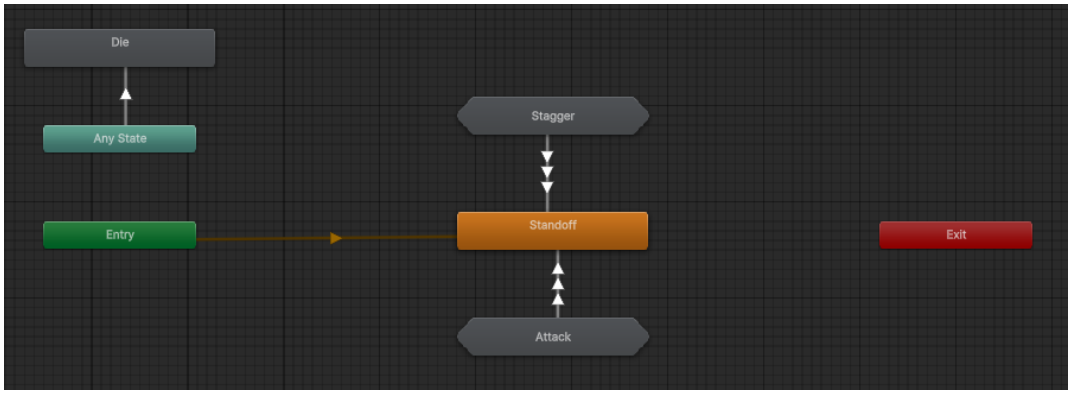
States

Player character



- **Standby&Move：** The default state, which can immediately transit to any other state
- **Dodge：** Can immediately transition from Standby&Move and Attack states
- **Attack：** This state transitions from the Standby&move state and has different interrupt levels depending on the attack moves.
- **Stagger：** It can transition from any state immediately when interrupted by an opponent's attack, exit until the end of the animation.

Enemy



- **Standoff：** The enemy's default state, can immediately transit to other states.
- **Attack：** This state can transition from the Standoff state and has different interrupt levels depending on the attack moves.
- **Stagger：** It can transition from any state immediately when interrupted by an opponent's attack, exit until the end of the animation.

Interrupt and anti-interrupt level

For both player character and enemy, when their attack interrupt level is higher than the opponent's current anti-interrupt Level, they will interrupt the opponent's current state and make them stagger.

Skills/Moves

Player character

Normal attack

There are five combos of normal attacks with both interrupt and anti-interrupt levels of 1.

Branch attack

Branch attacks are available as the next combo after the normal attack combo 2 (with sufficient energy), and it will cause a lot of damage, with both interrupt and anti-interrupt levels of 3.

Dodge

The player will be invulnerable when dodging and can flee from the threat with the movement.
Bullet Time: When the player dodges while an enemy is attacking and is in the enemy's attack range (in fact, the dodge detecting range is larger than the attack range), the bullet time effect will be triggered, during which the player will become invulnerable, and all player attacks will have the interrupt level of 3. And the enemy will lose targeting on you.

Enemy

Normal attack

The normal attack will cause less damage to the player, with only 0 anti-interrupt level and a 1 interrupt level, but the anticipation time is short than heavy attacks.

Heavy attack

The heavy attack deals high damage to the player and has an anti-interrupt level and interrupt level of 2, but the heavy attack has longer anticipation time and visual effects and sound effects as cues.

Jump heavy attack

This is a special version of the heavy attack that has the same stats as the regular heavy attack, but during the anticipation, the enemy will jump to the player's position, which means he can attack the player from far away.

Angry counterattack

It has the same animation and stats as the heavy attack, but it is triggered by the anger mechanism. When the enemy accumulates a certain amount of anger, he will immediately use the heavy attack.

Combat resources

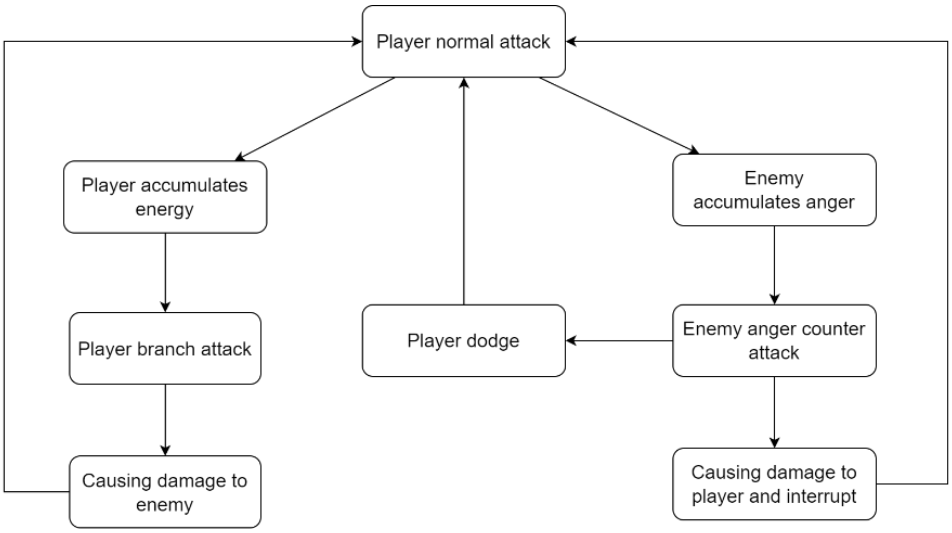
Energy (player)

Players can accumulate energy through normal attacks, and using branch attacks consumes energy.

Anger (enemy)

Enemies will accumulate anger when they take normal attacks from the player and will use the heavy attack immediately when they get a certain amount of anger. Anger will also increase the chance for the enemy to choose a heavy attack when attacking. And anger will reset to 0 on the enemy's next heavy attack.

Resource related combat loop



Ideas and thoughts on core mechanisms design

- According to the interrupt level set-up, the interrupt relationship for all moves is: **Player branch attack > enemy heavy attack > player normal attack > enemy normal attack > other states**.

This gives both the player and the enemy the opportunity to counter each other, but players have an overall advantage to keep control of the combat.

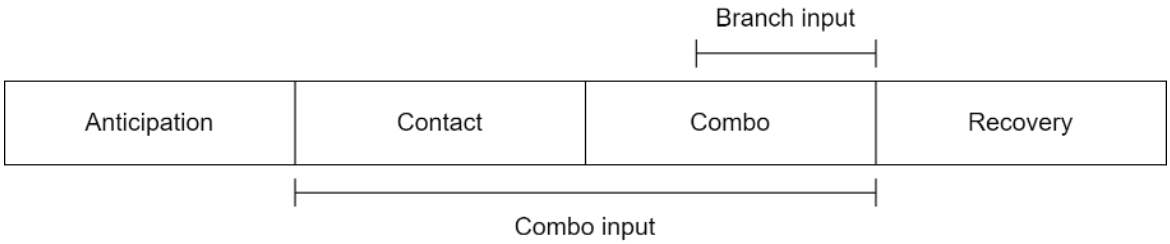
- The enemy's angry counterattack can prevent the player from keeping using normal attacks to suppress the enemy, bringing changes and risks to the combat.
- Player dodge can counter any enemy attack, including the normal attack and the heavy attack (including those triggered by the angry counterattack), so the player always has the opportunity to take control of the combat and take advantage.
- The accumulation of player energy is a positive feedback loop design, where the player has the opportunity to use branch attacks to cause a higher damage after taking the risk of normal attacking the enemy . It can increase the combat intensity and keep the player excited.
- The accumulation of enemy anger is also triggered by the player's normal attack, which is a negative feedback loop mechanism that serves as a balance and provides changes for the combat dynamics. However, it's a mechanism of the enemy, and the player will not feel the constraint directly. And the player can always use dodge as a countermeasure to take advantage.
- The accumulation of both resources is triggered by the player's behaviors, so the player controls combat pacing.

Controls and effects

State transitions

Attack combo

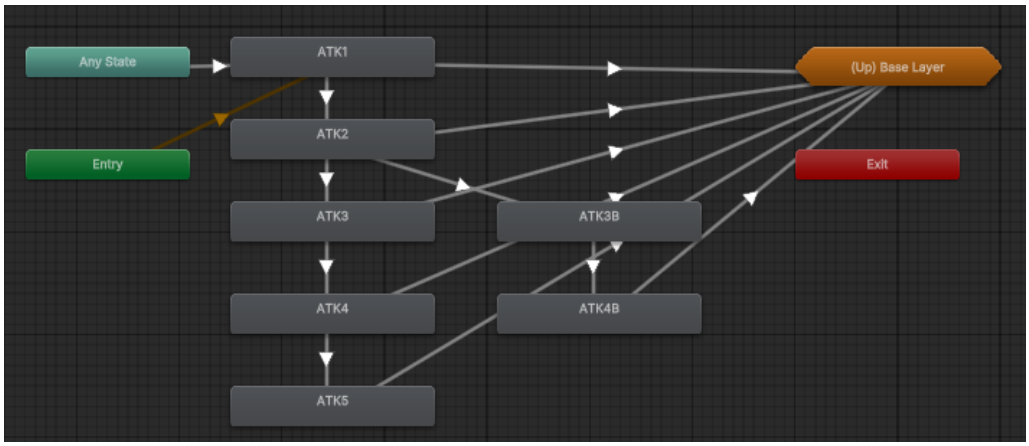
Every player's attack animation has four phases



- **Anticipation:** Preparation before the attack takes effect.
- **Contact:** The attack takes effct, and opponents within the attack range will be damaged or interrupted.
- **Combo ready:** If there is a new attack instruction, the it will immediately transition to the next attack combo.
- **Recovery:** If the player did not input attack instruction for the next combo or this is the final combo, the state will transition to Standby as default at this phase

Combo input time window: The start time for accepting the next attack combo input is usually earlier than the Combo ready phase, which allows the player to input combo instruction earlier for a smoothy and tolerant experience. Some call it input buffering.

Branch input time window: If the player inputs an attack instruction slightly later after the start of the combo ready phase in normal attack combo 2, the next combo will become a branch attack. Now, a combo tree with a branch can be implemented with just one button as input.



Dodge

Players can immediately transition to dodge (except from stagger and branch attack). This reduces the decision risk for the players.

Why dodge is not available during the branch attack? Because the player is invulnerable with branch attack, meaning there is no risk, and I don't want the players to lose the opportunity to attack due to misoperation and subconscious reactions.)

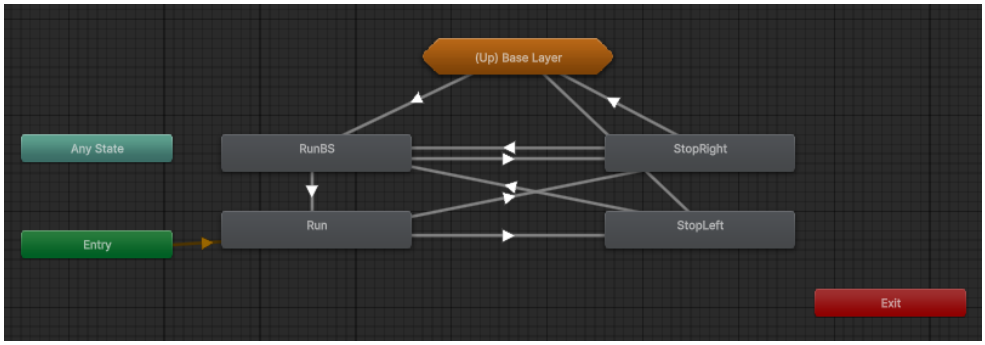
Other transition optimizations

Recovery phase of dodge and attack

During the transition from dodge and attack to standby, the recovery phase at the end of the animation can be interrupted by movement and new attack input, which can make the player feel smoother in control and more flexible in character actions. Strategically, it reduces the cost and risk of player behavior.

Trigger different run-to-stop animations based on the timing and pose of running

When the player runs, the game tracks and updates the current pose through animated events and selects proper run-to-stop animations for transitioning to standby.



Targeting system

Target selection

In the demo, I only implemented a simple strategy: selecting the nearest enemy within a certain distance as the target. However, I have more unimplemented strategies and designs, and I stated them at the end of the document.

Character rotation recentering

When the player initial attacks (Combo 0), the game will update the target selection, and the player character will rotate towards the target.

I didn't use the mechanism of "suck to target" because the attack animation had already moved the character forward.

Camera recentering

When the player initial attacks (Combo 0), the camera also smoothly rotates and points towards the target, but this only occurs when the camera direction is opposite to the direction to the target (>90 degrees), because I don't want the game to override camera controls too frequently to make players uncomfortable.

Feedback on the combat dynamics

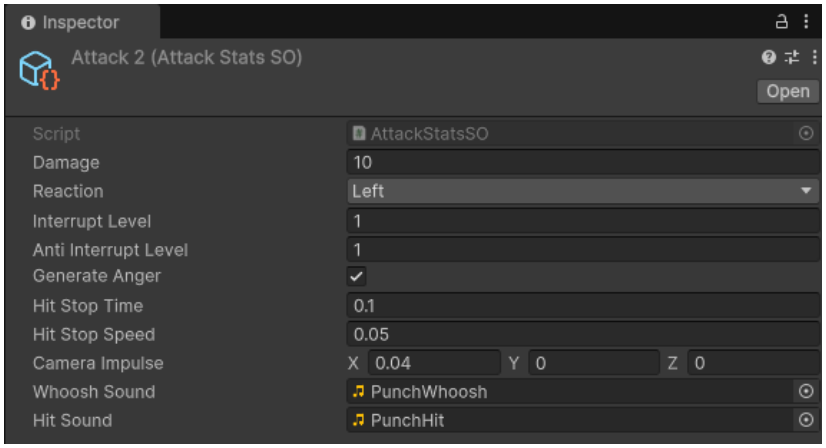
HUD

Player health bar & player energy bar & player dodge cooldown icon & enemy health bar

These are the basic information that players need to know to get access to the combat situation and make decisions.

Why not show the enemy anger bar? Because as I mentioned, the enemy's angry counterattack can provide more variety and possibilities for combat, and I didn't want the player to know in advance when the enemy will counterattack

Player Attack Feedback



Visual effects and sound effects

Visuals and sound effects are common ways to enhance the aesthetics of the action, and here I implemented not only the "whoosh" effect, but also the hit effect so that the player knows if they have successfully hit the target

Hit stop

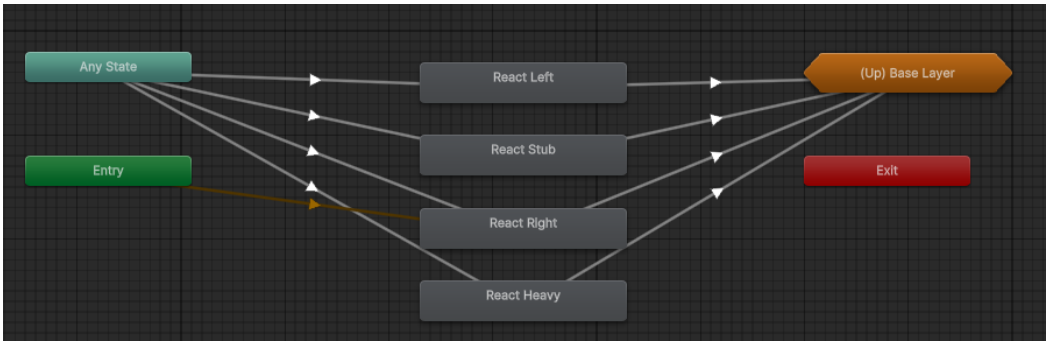
When the player character successfully hits an enemy, the animation slows down shortly to provide a sense of resistance from impact.

Camera shaking

Camera shaking in different directions and amplitude curves can provide feedback on the inertia and force of the impact.

Enemy reaction animation

The enemy will react with different animations for different player attacks. It can also provide feedback on the inertia and force of the impact.



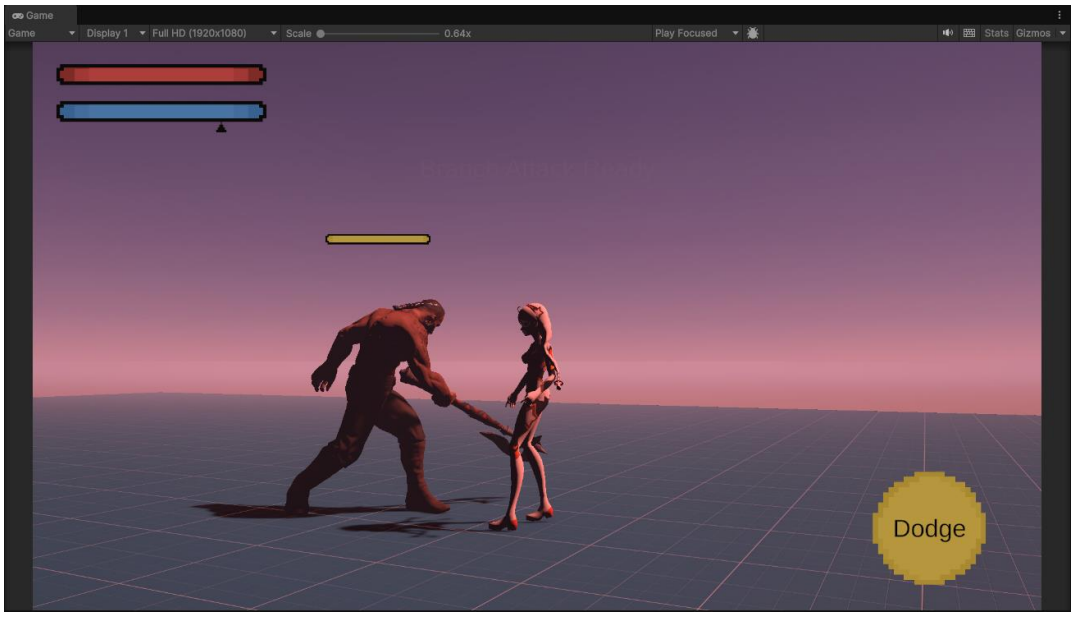
Damage indicator

When the player hits an enemy, a damage number will be generated on the screen. Although this is not a simulation of physical reactions, it is still a confirmation of the player's action and feedback on high-level cognition.

Combat information accessibility enhancement

Post-processing effect on the player receiving an attack

When a player got attacked, the screen briefly turns red to warn the player and remind him to pay attention to his health.



Post-processing effect on entering bullet time

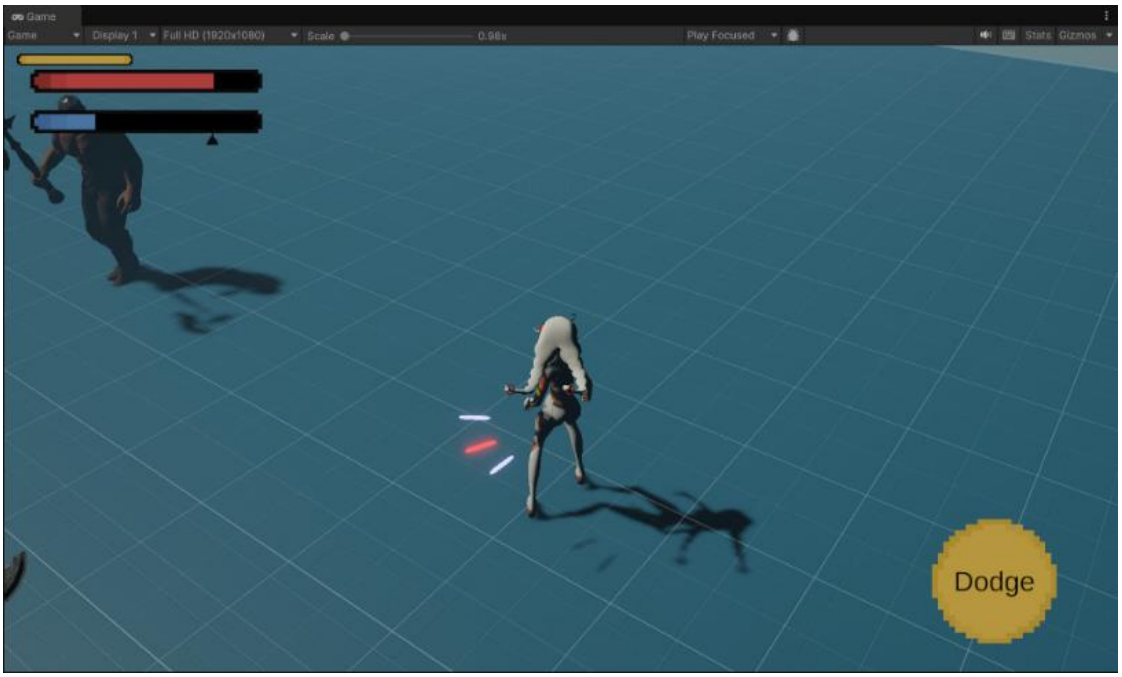
In addition to slowing down enemy animations, the player can also know the bullet time through special post-processing effects and decide to counterattack or flee from danger.

Branch attack input time window cue

When the player has enough energy to use the branch attack, there will be visual and sound effects as cues in the branch attack input time window of the normal attack combo 2.

Off-screen threat indicator

When the enemy is outside the screen, players can know the enemy's location and possible upcoming attacks through threat indicators around the character and take timely reactions.



Enemy heavy attack cue

In the anticipation phase of enemy heavy attack, there will be visual and sound effects as cues for the player to make timely responses.

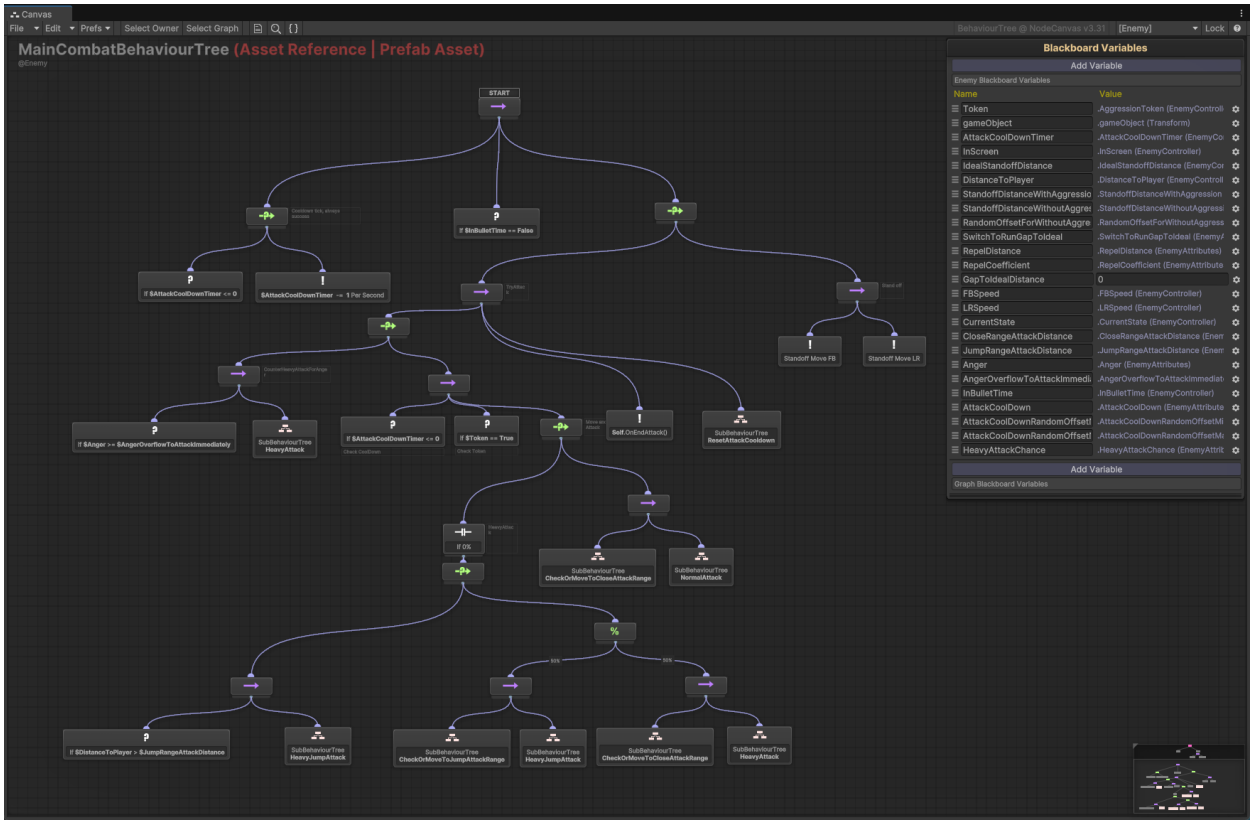
Ideas and thoughts on controls and effects design

In addition to combat aesthetics, I am more concerned about how the design of the above aspects will serve my design goal: providing an easy-to-learn action and friendly game experience for casual players. Therefore, my design is committed to achieving the following goals:

- **High accessibility of combat information:** Provide as many ways as possible to convey key combat information to players, such as upcoming threats and advantageous/disadvantageous situations
- **High input tolerance:** Expand the input window so that players can easily do the actions they want.
- **Low decision risk and cost:** The player's behavior and actions come at a cost, such as they can be attacked on executing incorrect actions or the recovery phase of actions. Transitioning early and immediately to another state(like dodge) can reduce the cost.

AI Design

In my demo, I mainly use behavior trees for enemy decision-making. Sorry, I am unable to show all the sub behavior trees in the document, as there are too many of them. Don't worry. I will describe the enemy's behavior in natural human language below.



Individual behaviors

Attack permission

Enemies in standoff state are always eager to attack, but they must meet two conditions to attack:

- **No Attack Cooldown**: Each enemy has its attack cooldown. This mechanic is designed to prevent an enemy from keeping attacking.
- **Acquire a Token from the enemies manager**: I'll explain it later.

Attack decisions

Normal attack or heavy attack

When enemies are allowed to attack, they will have the chance to use either a normal attack or a heavy attack. The choice is based on probability.

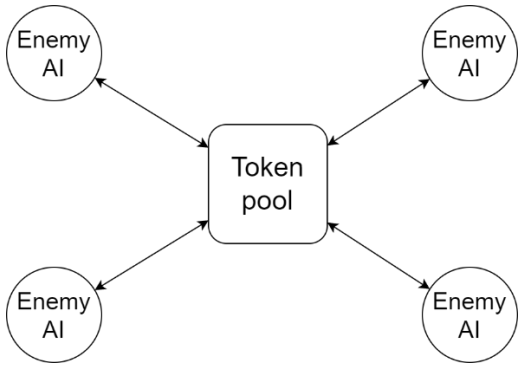
The probability of choosing a heavy attack = base probability + current anger.

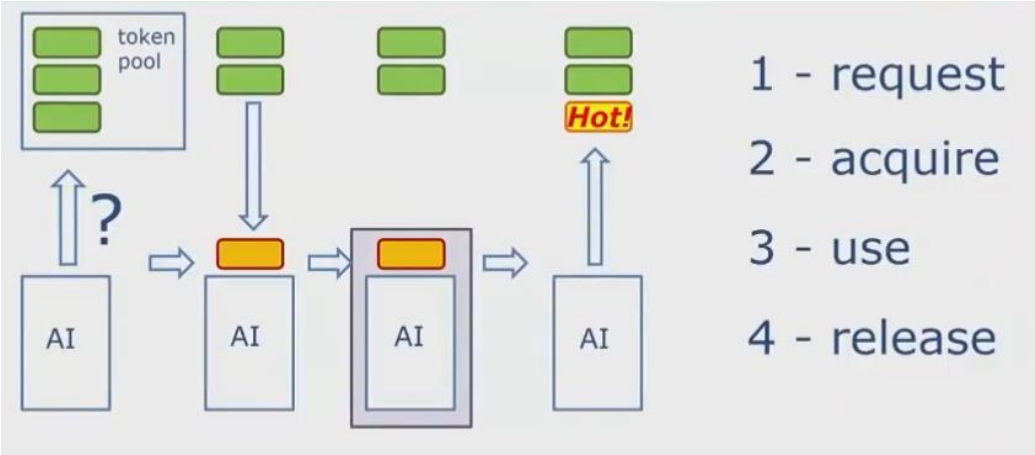
Heavy attack or jump heavy attack

When an enemy chooses a heavy attack, they will have the possibility to use a regular heavy attack or a jump heavy attack, depending on the enemy's current distance from the player. The farther away from the player, the more likely he is to choose jump heavy attack.

Group behaviors

Aggression token system





As I mentioned, the enemy needs a token for attack. He needs to acquire a token from the enemy manager, use it during attacks, and then release and return it. This is a centralized system driven by the enemy manager and a unique token pool.

Hot token

The newly released token is not available, and it needs to wait for a cooldown before it can be granted to another enemy.

Priority on granting tokens

When choosing which enemy to grant the token to, the Enemy Manager will consider the following conditions of the enemy in the order:

- Whether they are in screen.

- Whether they are on attack cooldown.
- Whether they are in stagger.

Steering behaviors in standoff

Longitudinal speed control

Because the enemy is always facing the player, his longitudinal movement changes the distance to the player.

There are two ideal distances: **aggressive distance** (when the enemy has a token but has not yet attacked) and **non-aggressive distance** (when the enemy does not have a token); the former is smaller than the latter.

The enemy will constantly adjust his longitudinal speed to reach the ideal distance, and the farther away he is, the faster he moves.

Lateral speed control

Collision avoidance (force-based)

Enemy lateral movement is mainly to prevent multiple enemies from congregating in the same position, or occlusion of each other.

Here, the enemy will move away from the nearest enemy by adjusting the lateral speed. Because the player's FOV is a cone area, the closer the enemy is to the player, the smaller the enemy's perception range.

(This is a combat design document, so I won't discuss my algorithm specifically. But if we have a chance to talk, I will be happy to discuss it)

Ideas and thoughts on AI design

My AI design serves two goals:

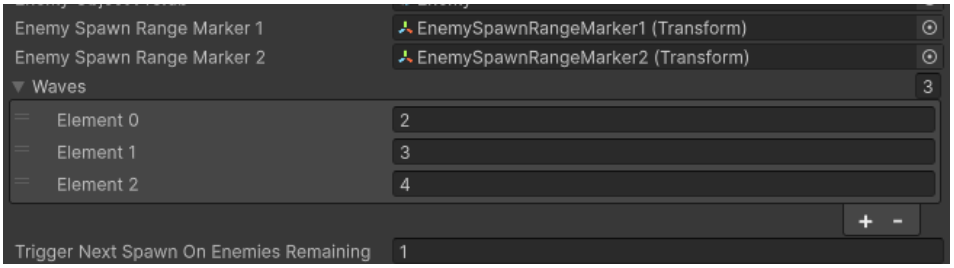
- **Decision-making focused on player behavior**: Enemy movements will be based on the player's position. When the enemy decides which attack to use, they will also consider the current anger generated by the player's previous normal attacks and the player's position.
- **Controllable combat intensity:** The size of the token pool and the cooldown time of the hot token determine the global combat intensity, that is, the frequency of enemy attacks and the number of aggressive enemies.

Game Mode Design

Wave-based enemy spawn strategy

In encounter design, we want the player to engage in a longer combat, but spawning all enemies at once can make the level too crowded and get the combat intensity out of control.

With the configuration of different waves, we can control the number of enemies in the level and the combat intensity.



Trigger next spawn on enemies remaining

We don't have to wait until all the enemies are destroyed to spawn the next wave of enemies, and we can spawn new enemies when there are only a few enemies left in the level to maintain the combat intensity.

Unimplemented improvements and future designs

More intelligent target selection

- Select the enemy within a certain angle as the target based on the direction of the current movement input.
- Leave the target selection unchanged after dodging or stagger for the player to counterattack

More combat camera designs

- During combat, the “look at point” shifts to the direction of the enemy, letting the player pay more attention to the enemy's actions.

- When the player fights a single enemy, the camera will use a smaller distance to the player (radius) and an appropriate horizontal (shoulder) offset, making it easier for the player to perceive the position and action details of the enemy and himself.

- When the player is fighting more enemies, the camera uses a larger distance and an appropriate top-down angle to make it easier for the player to see more enemies.

Movement ability

The player can hook and dash to the enemy, and one of the keys to designing this mechanic is to implement an aiming and targeting mechanic that is tolerant of precision.

More enemy archetypes

- **Ranged Enemies**: They can create threats from far away and force the player to move around the level to eliminate them.
- **Multiple-tier enemies**: A mix of elite and normal enemies, and even enemies with different styles and stats, will force the player to fight with more thought and strategy.

Online resources used

- **Enemy model & animations：** Mixamo https://www.mixamo.com/
- **Visual effects：** Unity Asset Store (Author: Kyeoms) https://assetstore.unity.com/publishers/52756
- **Sound effects:** Unity Asset Store (Punch and Fighting Sounds) https://assetstore.unity.com/audio/sound-fx
- **UI images：** OpenGameArt https://opengameart.org/
- **Behavior tree tool:** NodeCanvas https://nodecanvas.paradoxnotion.com/