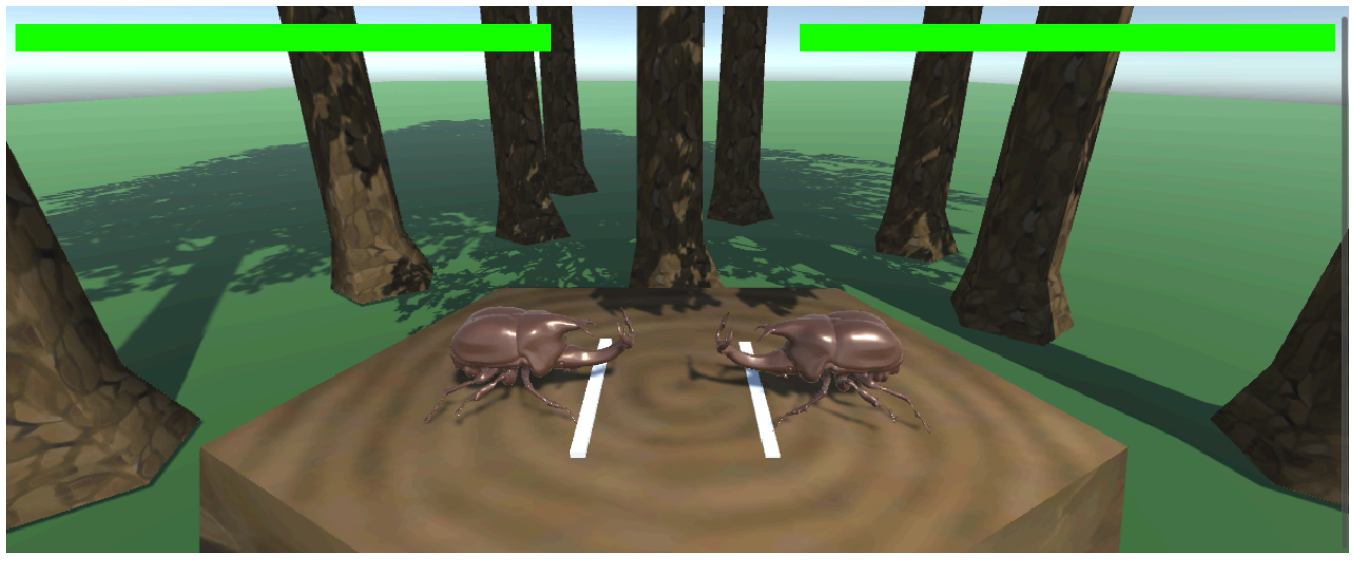
KonchuXKonchu



目次

制作概要

昆虫バトルx手押し相撲をコンセプトに制作した格闘ゲーム。
友人たちと共に、飲食店や射的などの屋台を出すお祭りイベントを定期的に開催しており、このイベントで子どもたちが楽しめるコンテンツとしてゲームを作成した。

本作品では、Nintendo SwitchのJoyconを利用した格闘ゲームである。プレイヤーはJoyconを上下に振る、手前に引くといった動作でカブトムシを操作する。上下に振ることで攻撃、手前に引くことで避けるアクションを行える。バトルをより面白くする要素として避けアクションが成功した時には次に繰り出す攻撃の攻撃力が増大する。

制作期間

2022年8月～製作中

プレイ環境

- Windows PC
- Nintendo Switch Joyconを利用

作品紹介動画

https://youtu.be/31UX2mh__RE

KonchuXKonchuプレイ動画

https://youtu.be/ChQI1wfwGh0

KonchuXKonchuゲーム内動画

作業範囲

一人で制作を行なった。カブトムシの動作や攻撃判定、Joyconの加速度からスイングを検知するスクリプトを作成、ゲームデザインやカブトムシモデルのアニメーション作成などを行なった。

機能実装（KabutomushiController.cs）

以下に、カブトムシの動作を記述したC#プログラムを示す。

現在のステート取得と避け成功判定

- Update()関数内で毎フレーム動作する処理
- 相手と自分のステートを取得し、特殊攻撃発動可能かどうかを判定する。
- カブトムシには、通常状態（Idle)、攻撃状態(Attack)、避け状態（Dodge）の三つのステートがある。

動作順序

1. 相手のカブトムシオブジェクトのanim_state変数に格納された相手のステートを表す文字列をopponent_stateに格納。
2. 自身のanim_state変数にも現在の自身のステートを表す文字列を格納。
3. 相手の状態が”Attack”中であり、自身の状態が”Dodge”中であれば、避け成功とみなして、特殊攻撃可能であることを示すspecial_attackableフラグをtrueにする。同時に、エフェクトと効果音を鳴らす。

```
opponent_state = GameObject.Find(opponent_player).GetComponent<KabutoController>().anim_state;

if (anim.GetCurrentAnimatorStateInfo(0).IsName("Idle"))
{
    anim_state = "Idle";
}
else if (anim.GetCurrentAnimatorStateInfo(0).IsName("Attack"))
{
    anim_state = "Attack";
}
else
{
```

```
    anim_state = "Dodge";
}

if ((opponent_state == "Attack") && (anim_state == "Dodge") && !special_attackable)
{
    special_attackable = true;
    special_effect.SetActive(true);
    audioSource.PlayOneShot(special_effect_sound);
}
```

Joyconの加速度取得とスイング判定

- Update()関数内で毎フレーム動作する処理
- Joyconの3軸それぞれの向きの加速度をライブラリJoyconLibの機能を用いて取得し、unityの座標軸に合わせて、格納。
- 上方向の加速度とスイング検知のための閾値、攻撃可能な状態かどうかを示すフラグを引数としてスイング判定を行うCheckSwing関数を呼び出す。

```
accel.x = (-1) * using_joycon.GetAccel().y;
accel.y = using_joycon.GetAccel().z;
accel.z = (-1) * using_joycon.GetAccel().x;

swing_accel_y = accel.y;
attack_flag = CheckSwing(swing_accel_y, accel_threshold_attack, attackable);
```

```
switch (attack_flag)
{
    // 一定の加速度以上の上スイングが行われ、かつattackableがtrueだったら攻撃
    case 1:
        if (anim_state == "Idle")
        {
            attackable = false;

            anim.SetBool("attackOn", true);
            if ((opponent_state == "Attack") || (opponent_state == "Idle"))
            {
                if (!special_attackable)
                {
                    attacked_value -= 0.02f;
                    UpdateGage(attacked_value);
                    audioSource.PlayOneShot(attack_sound);
                }
                else
                {
                    attacked_value -= 0.1f;
                    UpdateGage(attacked_value);
                    audioSource.PlayOneShot(special_attack_sound);
                    special_attackable = false;
                    special_effect.SetActive(false);
                }
            }
            // 相手のHPが０になったら
            if (attacked_value <= 0f)
            {
                attacked_value = 1f;
            }
```

```
            dodge_flag = 0;
        }
        break;
    // 一定以上の加速度で上スイング中だが一定時間を超えていないときは
    // 上スイング継続時間をプラス
    case 2:
        attackable = true;
        anim.SetBool("attackOn", false);
        break;
    case 0:
        break;
    default:
        Debug.Log("Exception");
        break;
}
```

ソースコード

https://github.com/enbas0721/KonchuXKonchu