PROGRESS REPORT

FINAL PROJECT: Top-Down Hack N' Slash

Andrew Adame

ID: 007516100

LEGEND: COMPLETED – UNFINISHED – WIP – FIX – FIXED

GITHUB: https://github.com/andrewadame/UnityProjectsCSE-4410/tree/master/FinalProject

UNITY PLAY: https://play.unity.com/mg/other/finalprojectcse4410

1. Create new project FinalProject
2. Create folders containing important assets (scripts, prefabs, animation, etc)
3. Create a Top-Down Hack N' Slash
    a. Design Level
        i. Tilemap
        ii. Camera
            1. CmCtrlr
                a. A camera that can follow the player
    b. Player
        i. Sprite
            1. Amalgam
        ii. Animation
            1. Idle
            2. Walking
            3. Attacking
            4. Dodging
                a. Added animation
            5. Taking Damage
        iii. Behavior
            1. Components
                a. BoxCollider2D
                b. Rigidbody2D
            2. Attack
                a. Use box collider to damage enemy
        iv. Dodge (new mechanic #1)
            1. Pressing 'Space" will allow players to dodge attacks, giving them iframes to keep from taking damage
        v. Leveling System

1. Player levels up after gaining certain amount of exp
2. Gets stronger every level
3. Exp requirement gets higher every level

vi. Scripts

1. PlyrCtrlr
2. PlyrMelCol

c. Enemy

i. Follows player until within attack range

1. Uses a box collider to deal damage to player

ii. Sprites

1. BnC
    a. Made by Penusbmic
2. Gnr
    a. Made by Penusbmic

iii. Scripts

1. EnCtrlr
2. EnMelCol

d. Visuals

i. Player sprite and animations made by me

ii. Tile assets made by Pupkin

1. Link to asset pack: https://trevor-pupkin.itch.io/tech-dungeon-roguelite
2. Link to their itch.io: https://trevor-pupkin.itch.io

iii. Enemy sprites and enemy animations by Penusbmic

1. Link to the asset pack: https://penusbmic.itch.io/sci-fi-character-pack-12
2. Link to their itch.io: https://penusbmic.itch.io

e. Gameplay

i. Objective

1. Survive incoming waves of enemies and upgrade yourself

ii. Game Start

1. Instruction screen before game start
2. Player spawns in starting area to buy items
    a. Starting area serves as a shop/leveling stop after every wave
3. Wave I begins once player crosses red lines
4. Enemy engages player. Player gains a set amount exp and money when enemy is killed
5. When player reaches the required exp to level up, they do so with next requirement increasing

6. After set amount of enemies killed, wave ends and enemies stop spawning
    a. Player has 30 seconds till next wave

iii. Shop and Chest System
1. Create a shop
2. Create a chest
3. Scripts

iv. Wave System (new mechanic #2)
1. Certain amount of enemies spawn at a specific rate
2. After all enemies are defeated, next wave begins
3. A certain amount of time to rest between each wave

v. Game Over
1. When player's health drops to 0, player dies
2. Game over screen will display, allowing player to restart

vi. Scripts
1. GameCtrlr

f. UI

i. Player UI
1. Health bar that decreases over time
2. Display money, experience points, level, and exp requirement to level up
3. Script
    a.

g. **EXTRA**

i. Audio

ii. Inventory
1. Items
    a. Healing Item
        i. Power core?
    b. Weapons/Equipment
        i. Acts as modifiers for damage/speed/health
2. Can pick up to three items, adds to inventory
3. Can use items by flipping through inventory
4. Will be displayed in bottom left of screen

iii. Improved Leveling System
1. Allow player to choose what stat they want to upgrade
2. They can do this after every wave

SCRIPTS

PlyrCtrlr

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class PlyrCtrlr : MonoBehaviour
{
    Rigidbody2D plyrRgdBdy;
    Vector2 input;
    public float spd;
    public float dgeSpd;
    Vector2 dgeVec; //Dodge direction

    Animator anim;
    SpriteRenderer rend;

    int lkDir = 0;  //0=down, 1=left/right, 2=up
    bool mvng = false;

    public float mxHlth;
    public float hlth;
    public Image htlhUI;

    public int mxMny;
    public int mny;
    public Text mnyTxt;

    public float attack;
    public int level = 1;
    public float exp;
    public float expToNxt;
    public AnimationCurve expCurve = new AnimationCurve();
    public Text expTxt;

    public float ifrmeTme = 0.6f;
    float iframe;
    public float dgeTme = 0.8f;
    float dge;

    public GameObject meleeCol;

    private void Awake()
    {
        plyrRgdBdy = GetComponent<Rigidbody2D>();
        anim = GetComponent<Animator>();
        rend = GetComponent<SpriteRenderer>();
        expToNxt = CalcExp(level);
        for (int i = 1; i <= 30; i++)
        {
            expCurve.AddKey(i, CalcExp(i));
        }

        hlth = mxHlth;
        mny = mxMny;
    }

    void Start()
    {
```

```
}

// Update is called once per frame
void Update()
{
    if (iframe > 0)
    {
        iframe -= Time.deltaTime;
    }
    if(dge > 0)
    {
        dge -= Time.deltaTime;
    }

    input = new Vector2(Input.GetAxis("Horizontal"), Input.GetAxis("Vertical"));

    plyrRgdBdy.AddForce(input * spd * Time.deltaTime);

    mvng = (input.x != 0 || input.y != 0);


    if (input.y < 0)
    {
        lkDir = 0;

        //Attack Down
        meleeCol.transform.localPosition = new Vector3(0, -0.16f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.x > 0)
    {
        lkDir = 1;

        //Attack Right
        rend.flipX = false;
        meleeCol.transform.localPosition = new Vector3(0.111f, -0.003f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.y > 0)
    {
        lkDir = 2;

        //Attack Up
        meleeCol.transform.localPosition = new Vector3(0, 0.16f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.x < 0)
    {
        lkDir = 1;

        //Attack Left
        rend.flipX = true;
        meleeCol.transform.localPosition = new Vector3(-0.111f, -0.003f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
```

```
        anim.SetInteger("dir", lkDir);
        anim.SetBool("mov", mvng);


        //Dodge
        ///////////
        if (Input.GetKeyDown(KeyCode.F) && input != Vector2.zero)
        {
            Dodge();
        }
        ///////////


        if (Input.GetKeyDown(KeyCode.Space))
        {
            SwgAtk();
            Debug.Log("Attack!");
        }

        //EXP TEXT
        if (Input.GetKeyDown(KeyCode.J))
        {
            AddExp(20);
        }

        htlhUI.fillAmount = hlth / mxHlth;
        mnyTxt.text = "Coins: " + mny.ToString();
        expTxt.text = "Level " + level.ToString() + " - Exp: " + exp.ToString() + "/" +
expToNxt.ToString();

    }
    public void SwgAtk()
    {
        anim.SetBool("atk", true);
        Invoke("RstAtk", 0.1f);
    }

    void RstAtk()
    {
        anim.SetBool("atk", false);
    }

    public void Heal(float amt)
    {
        hlth += amt;
        if (hlth > mxHlth)
        {
            hlth = mxHlth;
        }
    }

    public void Dodge()
    {
        if (iframe <= 0 && dge <= 0)
        {
            dgeVec = input.normalized;
            iframe = ifrmeTme;
            dge = dgeTme;
```

```
            //Dodge!
            plyrRgdBdy.velocity = dgeVec.normalized * dgeSpd;
            Debug.Log("Dodge!");
        }
    }

    public void Damage(float amt)
    {
        if (iframe <= 0)
        {
            hlth -= amt;
            iframe = ifrmeTme;
            if (hlth <= 0)
            {
                Die();
            }
        }
        /*
        if(hlth <= 0)
        {
            Die();
        }
        */
    }

    public void Die()
    {
        gameObject.SetActive(false);
        FindObjectOfType<GmeCtrlr>().gmeOvr = true;
        FindObjectOfType<GmeCtrlr>().gameOverUI.SetActive(true);
        Time.timeScale = 0;

    }

    public void AddMny(int amnt)
    {
        mny += amnt;
    }

    public float CalcExp(int level)
    {
        float expNded;
        expNded = level * 50f;
        return expNded;
    }

    public void AddExp(float amt)
    {
        exp += amt;

        if (exp >= expToNxt)
        {
            LevelUp();
        }
    }

    public void LevelUp()
```

```
    {
        level++;
        exp -= expToNxt;
        attack = attack + 5f;
        spd = spd + 50f;
        mxHlth = mxHlth + 10f;
        Heal(mxHlth);
        expToNxt = CalcExp(level);
    }
}
```

PlyrMelCol

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlyrMelCol : MonoBehaviour
{
    PlyrCtrlr plyr;

    private void Awake()
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        //If hitbox hits enemy, damage
        if (collision.gameObject.CompareTag("Enemy"))
        {
            collision.gameObject.GetComponent<EnCtrlr>().Dmg(plyr.attack);
            Debug.Log("Damage!");
        }
    }

    private void OnTriggerStay2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            collision.gameObject.GetComponent<EnCtrlr>().Dmg(plyr.attack);
        }
    }
}
```

GmeCtrlr

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class GmeCtrlr : MonoBehaviour
{
    public GameObject gmeStrtUI;
    public bool gmeStrt;
    public GameObject gameOverUI;
```

```
    public bool gmeOvr;

    // Start is called before the first frame update
    void Start()
    {
        gmeOvr = false;
        gmeStrt = false;
    }

    // Update is called once per frame
    void Update()
    {
        if (!gmeStrt)
        {
            StartScreen();
        }

        if (gmeOvr && Input.anyKeyDown)
        {
            Restart();
        }
    }

    void Restart()
    {
        SceneManager.LoadScene("SampleScene");
        Time.timeScale = 1f;
    }

    void StartScreen()
    {
        gmeStrtUI.SetActive(true);
        Time.timeScale = 0;

        if(Input.anyKeyDown)
        {
            gmeStrt = true;
            gmeStrtUI.SetActive(false);
            Time.timeScale = 1f;
        }

    }
}
```

WaveCtrlr

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
[System.Serializable]

public class Wave
{
    public string wvNme;
    public int nmOfEn;
    public GameObject[] enType;
    public float spwnIntrvl;
}
```

```
public class WaveCtrlr : MonoBehaviour
{
    public Wave[] waves;
    public Transform[] spwnPnt;

    private Wave crntWve;
    private int crntWveNm;
    private float nxtSpwnTmer;
    public Text wveNm;

    private bool cnSpwn = true;

    private void Update()
    {
        crntWve = waves[crntWveNm];
        spwnWve();

        //Find amount of enemys on map
        GameObject[] totEn = GameObject.FindGameObjectsWithTag("Enemy");
        if(totEn.Length == 0 && !cnSpwn && crntWveNm+1 != waves.Length)
        {
            crntWveNm++;
            cnSpwn = true;
        }

        wveNm.text = "Wave: " + crntWveNm.ToString();

    }

    void spwnWve()
    {
        if (cnSpwn && nxtSpwnTmer < Time.time)
        {
            GameObject rndEn = crntWve.enType[Random.Range(0, crntWve.enType.Length)];
            Transform rndPnt = spwnPnt[Random.Range(0, spwnPnt.Length)];
            Instantiate(rndEn, rndPnt.position, Quaternion.identity);
            crntWve.nmOfEn--;

            nxtSpwnTmer = Time.time + crntWve.spwnIntrvl;

            if(crntWve.nmOfEn == 0)
            {
                cnSpwn = false;
            }
        }
    }


}
```

CamCtrlr

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CamCtrlr : MonoBehaviour
{
```

```
    public Transform trgt;
    public float lrpSpd;

    Vector3 tempPos;
    [SerializeField]
    float minX, minY, maxX, maxY;

    // Update is called once per frame
    void FixedUpdate()
    {
        if (trgt == null) return;
        tempPos = trgt.position;
        tempPos.z = -10;


        //MIN
        if (trgt.position.x < minX)
        {
            tempPos.x = minX;
        }
        if (trgt.position.y < minY)
        {
            tempPos.y = minY;
        }

        //MAX
        if (trgt.position.x > maxX)
        {
            tempPos.x = maxX;
        }
        if (trgt.position.y > maxY)
        {
            tempPos.y = maxY;
        }

        transform.position = Vector3.Lerp(transform.position, tempPos, lrpSpd *
Time.deltaTime);
    }
}
```

EnCtrlr

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class EnCtrlr : MonoBehaviour
{
    public float mxHp;
    [SerializeField]
    float hp;
    public float exp;
    public int mny;

    PlyrCtrlr plyr;
    public float iframeTme = 0.3f;
    float iframe;

    public enum enState { chase, atk };
```

```
public enState crntState;

Animator anim;
Rigidbody2D enRgdBdy;
//GmeCtrlr cont;

public float tmeBtwnAtk = 1f;
float cools;

public float spd;
int dir;

SpriteRenderer rend;
public float atkRng;
float dist;

public GameObject meleeCol;

private void Awake()
{
    plyr = FindObjectOfType<PlyrCtrlr>();
    //cont = FindObjectOfType<GmeCtrlr>();
    anim = GetComponent<Animator>();
    enRgdBdy = GetComponent<Rigidbody2D>();
    rend = GetComponent<SpriteRenderer>();

    hp = mxHp;
    iframe = iframeTme;
    crntState = enState.chase;
}

public void Dmg(float amt)
{
    if (iframe <= 0)
    {
        hp -= amt;

        if (hp <= 0)
        {
            Die();
        }
    }
}

void Die()
{
    gameObject.SetActive(false);
    plyr.AddExp(exp);
    plyr.AddMny(mny);
}
void Start()
{

}

// Update is called once per frame
void Update()
{
```

```
    if (iframe > 0)
    {
        iframe -= Time.deltaTime;
    }
    if (cools > 0)
    {
        cools -= Time.deltaTime;
    }

    switch (crntState)
    {
        case (enState.chase):
            Chase();
            break;
        case (enState.atk):
            Attack();
            break;
    }
    anim.SetInteger("dir", dir);
}

void Chase()
{
    dist = Vector2.Distance(transform.position, plyr.transform.position);

    if (plyr.transform.position.y < transform.position.y)
    {
        dir = 0;
        //meleeCol.transform.localPosition = new Vector3(0, -0.311f, 0);
        //meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (plyr.transform.position.x > transform.position.x)
    {
        dir = 1;
        rend.flipX = false;
        meleeCol.transform.localPosition = new Vector3(-0.1083f, 0, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (plyr.transform.position.x < transform.position.x)
    {
        dir = 1;
        rend.flipX = true;
        meleeCol.transform.localPosition = new Vector3(-0.55f, 0, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (plyr.transform.position.y > transform.position.y)
    {
        dir = 2;
        //meleeCol.transform.localPosition = new Vector3(0, 0.311f, 0);
        //meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }

    if (dist > atkRng)
    {
        Vector3 direction = plyr.transform.position - transform.position;
        enRgdBdy.AddForce(direction * spd * Time.deltaTime);
    }
    else
```

```
        {
            if (cools <= 0)
            {
                crntState = enState.atk;
            }
        }
    }

    void Attack()
    {
        anim.SetTrigger("atk");
        cools = tmeBtwnAtk;
        crntState = enState.chase;

    }
}
```

EnMelCol

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class EnMelCol : MonoBehaviour
{
    PlyrCtrlr plyr;

    private void Awake()
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        //If hitbox hits enemy, damage
        if (collision.gameObject.CompareTag("Player"))
        {
            collision.gameObject.GetComponent<PlyrCtrlr>().Damage(plyr.attack);
            Debug.Log("Player Damaged!");
        }
    }

    private void OnTriggerStay2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Player"))
        {
            collision.gameObject.GetComponent<PlyrCtrlr>().Damage(plyr.attack);
        }
    }
}
```