PROGRESS REPORT

MICROGAME #7: Action Game

Andrew Adame

ID: 007516100

LEGEND: COMPLETED – UNFINISHED – WIP – FIX – FIXED

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

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

1. Create new project Action Game Project
2. Create folders containing important assets (scripts, prefabs, animation, etc)
3. Create a basic Action Game
    a. Design Level
        i. Tilemap
            1. Provided by professor
        ii. Camera
            1. CmCtrlr
                a. A camera that can follow the player
    b. Player
        i. Sprite
            1. Barbarian
                a. Provided by professor
        ii. Behavior
            1. Components
                a. BoxCollider2D
                b. Rigidbody2D
        iii. Inventory
            1. Items
                a. Sword
                b. BloodPotion
                c. Colt
            2. Can pick up to three items, adds to inventory
            3. Can use items by flipping through inventory
                a. Get an ArgumentOutOfRange error when pressing the 'Use' button without an item equipped
            4. Has an unarmed attack that deals less damage than an equipped weapon

iv. Scripts
    1. PlyrCtrlr
    2. PlyrMelCol
    3. Invtry
    4. Item
    5. MelWpn
    6. RngWpn
    7. PrctleCtrlr

c. NPCs
    i. Non-hostile entities that can be interacted with
    ii. Two NPCs with a little dialogue when clicked on
        1. Dialogue moves on to other parts of the conversation when clicking on NPC
            a. A simple dialogue check should be able to fix it
    iii. Sprites
        1. Slime
            a. Provided by professor
    iv. Scripts
        1. NPCCtrlr
        2. Dial
        3. DialCtrlr

d. Enemy
    i. Follows player until within attack range
        1. Uses a box collider to deal damage to player
    ii. Sprites
        1. Slime
            a. Provided by professor
    iii. Scripts
        1. EnCtrlr
        2. EnMelCol

e. Visuals
    i. All sprites used were provided by the professor

f. Gameplay
    i. Game Start
        1. Player spawns on map
        2. Enemy spawns on map
        3. Chest and Shop items are initialized
        4. Enemy engages player. Player gains a set amount exp and money when enemy is killed
            a. Make a small wave system for enemies to spawn

5. When player reaches the required exp to level up, they do so with next requirement increasing

ii. Objective
1. Survive, technically. No particular objective made. Will be using this project as a proof of concept for the Final.

iii. Shop and Chest System
1. Create a shop
a. When player approaches shop, three random items of various rarities are displayed for the player to buy
b. Items display green if they can be bought, red if not
2. Create a chest
a. If player touches chest, chest will drop three items of various rarities
b. Player can pick them up and use
3. Rarity System
a. Five rarities each with different drop rates
i. Common: 50%
ii. Uncommon: 20%
iii. Rare: 15%
iv. Epic: 10%
v. Legendary: 5%
4. Scripts
a. Shop
b. ShopItem
c. Chest
d. ChestItem

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

v. Scripts
1. GameCtrlr

g. UI

i. Dialogue System
1. Player can click on an NPC to interact with them.
2. Script
a. Dial
b. DialCtrlr

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

3. Script
    a. DialCtrlr

h. **EXTRA**
    i. Audio

SCRIPTS

GmeCtrlr

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

public class GameCtrlr : MonoBehaviour
{

    public List<ShopItem> shpItms = new List<ShopItem>();

    public List<ChestItem> chstItms = new List<ChestItem>();

    public List<ChestItem> cmnItms = new List<ChestItem>();
    public List<ChestItem> uncmnItms = new List<ChestItem>();
    public List<ChestItem> rareItms = new List<ChestItem>();
    public List<ChestItem> epicItms = new List<ChestItem>();
    public List<ChestItem> lgndItms = new List<ChestItem>();

    void Awake()
    {
        for(int i = 0; i < chstItms.Count; i++)
        {
            switch(chstItms[i].item.itmRarity)
            {
                case Item.rairty.common:
                    cmnItms.Add(chstItms[i]);
                    break;
                case Item.rairty.uncommon:
                    uncmnItms.Add(chstItms[i]);
                    break;
                case Item.rairty.rare:
                    rareItms.Add(chstItms[i]);
                    break;
                case Item.rairty.epic:
                    epicItms.Add(chstItms[i]);
                    break;
                case Item.rairty.legendary:
                    lgndItms.Add(chstItms[i]);
                    break;
            }
        }

    }

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

```
    }

    public ShopItem GtRndItm(List<ShopItem> l)
    {
        int index = Random.Range(0, l.Count);
        ShopItem item = l[index];
        return item;

    }

    public ChestItem GtRndItm(List<ChestItem> l)
    {
        int index = Random.Range(0, l.Count);
        ChestItem item = l[index];
        return item;
    }
}
```

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);
    }
}
```

PlyrCtrlr

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

public class PlyrCtrlr : MonoBehaviour
{
    Rigidbody2D plyrRgdBdy;
    Vector2 input;
    public float spd;

    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 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;
    }

    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;
        meleeCol.transform.localPosition = new Vector3(0, -0.347f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.x > 0)
    {
        lkDir = 1;
        rend.flipX = false;
        meleeCol.transform.localPosition = new Vector3(0.3f, 0.2f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.y > 0)
    {
        lkDir = 2;
        meleeCol.transform.localPosition = new Vector3(0, 0.347f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }
    else if (input.x < 0)
    {
        lkDir = 1;
        rend.flipX = true;
        meleeCol.transform.localPosition = new Vector3(-0.3f, 0.2f, 0);
        meleeCol.transform.localScale = new Vector3(1, 1, 1);
    }

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

    if (Input.GetKeyDown(KeyCode.Space))
    {
        SwgAtk();
    }

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

    public void Die()
    {
        gameObject.SetActive(false);
        Time.timeScale = 0;
    }
```

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

    public float CalcExp(int level)
    {
        float expNded;
        expNded = level * 100f;
        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 (collision.gameObject.CompareTag("Enemy"))
        {
            collision.gameObject.GetComponent<EnCtrlr>().Dmg(plyr.attack);
        }
    }

    private void OnTriggerStay2D(Collider2D collision)
    {
```

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

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;
    GameCtrlr 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<GameCtrlr>();
        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(0.3f, 0.2f, 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(-0.3f, 0.2f, 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
{
    public float atk;
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if(collision.gameObject.CompareTag("Player"))
        {
            collision.gameObject.GetComponent<PlyrCtrlr>().Damage(atk);
        }
    }

    private void OnTriggerStay2D(Collider2D collision)
```

```
    {
        if(collision.gameObject.CompareTag("Player"))
        {
            collision.gameObject.GetComponent<PlyrCtrlr>().Damage(atk);
        }
    }
}
```

NPCCtrlr

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

public class NPCCtrlr : MonoBehaviour
{
    DialCtrlr diaCtrlr;
    public Dial[] dlgs;
    int crntDia = 0;

    private void Awake()
    {
        diaCtrlr = FindObjectOfType<DialCtrlr>();
    }

    private void OnMouseDown()
    {
        diaCtrlr.StrtDial(dlgs[crntDia]);
        crntDia = (crntDia + 1) % dlgs.Length;
    }
}
```

Dial

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

[System.Serializable]
public class Dial
{
    public string npcNme;
    public string[] dial;
}
```

DialCtrlr

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

public class DialCtrlr : MonoBehaviour
{
    Dial crntDl;

    public Text nmeUI;
    public Text dialUI;
    public GameObject UIPrnt;
```

```
    int crntIdx;

    public void StrtDial(Dial d)
    {
        crntDl = d;
        UIPrnt.SetActive(true);
        crntIdx = 0;
        nmeUI.text = crntDl.npcNme;
        dialUI.text = crntDl.dial[crntIdx];

    }

    public void NxtLne()
    {
        crntIdx++;
        if(crntIdx < crntDl.dial.Length)
        {
            nmeUI.text = crntDl.npcNme;
            dialUI.text = crntDl.dial[crntIdx];
        }
        else
        {
            extDial();
        }
    }

    public void extDial()
    {
        dialUI.text = "";
        nmeUI.text = "";
        UIPrnt.SetActive(false);
        crntIdx = 0;
    }
}
```

Shop

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

public class Shop : MonoBehaviour
{
    public float intrctDist;
    [SerializeField]
    float dist;

    public GameObject shpPrnt;
    PlyrCtrlr plyr;

    GameCtrlr cont;

    // Start is called before the first frame update
    void Awake()
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
        cont = FindObjectOfType<GameCtrlr>();
        PopShop();
    }

    // Update is called once per frame
```

```
    void Update()
    {
        dist = Vector2.Distance(transform.position, plyr.transform.position);
        if(dist <= intrctDist)
        {
            shpPrnt.SetActive(true);
        }
        else
        {
            shpPrnt.SetActive(false);
        }
    }

    public void PopShop()
    {
        ShopItem shpItm;
        for (int i = 0; i < 3; i++)
        {
            //shpItm = Instantiate(cont.shpItms[i]);
            shpItm = Instantiate(cont.GtRndItm(cont.shpItms));
            shpItm.transform.SetParent(shpPrnt.transform);
            shpItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);

        }

    }
}
```

ShopItem

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

public class ShopItem : MonoBehaviour
{
    public Item item;
    PlyrCtrlr plyr;
    Invtry invtry;
    SpriteRenderer rend;
    Text itmTxt;

    private void Awake()
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
        invtry = FindObjectOfType<Invtry>();
        rend = GetComponent<SpriteRenderer>();
        itmTxt = GetComponentInChildren<Text>();

        rend.sprite = item.itmSprte;

    }

    public void BuyItm()
    {
        if(plyr.mny >= item.itmCst)
        {
            plyr.AddMny(-item.itmCst);
```

```
            invtry.AddItem(item);
            Destroy(gameObject);
        }

    }

    private void OnMouseDown()
    {
        BuyItm();
    }

    private void Update()
    {
        itmTxt.text = item.itmNme + "\n" + item.itmCst;
        itmTxt.color = plyr.mny > item.itmCst ? Color.green : Color.red;
    }

}
```

Chest

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

public class Chest : MonoBehaviour
{

    public GameObject chstPrnt;
    PlyrCtrlr plyr;

    GameCtrlr cont;

    bool populated = false;
    void Awake()
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
        cont = FindObjectOfType<GameCtrlr>();
        PopChst();
    }

    public void PopChst()
    {
        ChestItem chstItm;
        for (int i = 0; i < 3; i++)
        {
            int r = Random.Range(0, 100);
            if(r < 3 && cont.lgndItms.Count != 0) //Legendary
            {
                Debug.Log("Legendary!");
                chstItm = Instantiate(cont.GtRndItm(cont.lgndItms));
                chstItm.transform.SetParent(chstPrnt.transform);
                chstItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);
            }
            else if (r < 10 && cont.epicItms.Count != 0) //Epic
            {
                Debug.Log("Epic!");
                chstItm = Instantiate(cont.GtRndItm(cont.epicItms));
                chstItm.transform.SetParent(chstPrnt.transform);
```

```
                chstItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);
            }
            else if (r < 30 && cont.rareItms.Count != 0) //Rare
            {
                Debug.Log("Rare!");
                chstItm = Instantiate(cont.GtRndItm(cont.rareItms));
                chstItm.transform.SetParent(chstPrnt.transform);
                chstItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);
            }
            else if (r < 50 && cont.uncmnItms.Count != 0) //Uncommon
            {
                Debug.Log("Uncommon!");
                chstItm = Instantiate(cont.GtRndItm(cont.uncmnItms));
                chstItm.transform.SetParent(chstPrnt.transform);
                chstItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);
            }
            else if (cont.cmnItms.Count != 0)//Common
            {
                Debug.Log("Common!");
                chstItm = Instantiate(cont.GtRndItm(cont.cmnItms));
                chstItm.transform.SetParent(chstPrnt.transform);
                chstItm.transform.localPosition = new Vector3((i * 0.5f) - 0.5f, 0, 0);
            }
            else
            {
                Debug.Log("No items in chest!");
            }

        }

    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        if(!populated && collision.gameObject.CompareTag("Player"))
        {
            PopChst();
            populated = true;
        }
    }
                                            }
```

ChestItem

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

public class ChestItem : MonoBehaviour
{
    public Item item;
    PlyrCtrlr plyr;
    Invtry invtry;
    SpriteRenderer rend;
    Text itmTxt;

    private void Awake()
```

```
    {
        plyr = FindObjectOfType<PlyrCtrlr>();
        invtry = FindObjectOfType<Invtry>();
        rend = GetComponent<SpriteRenderer>();
        itmTxt = GetComponentInChildren<Text>();

        rend.sprite = item.itmSprte;

    }

    public void PckUpItm()
    {
        invtry.AddItem(item);
        Destroy(gameObject);
    }

    private void OnMouseDown()
    {
        PckUpItm();
    }

    private void Update()
    {
        itmTxt.text = item.itmNme;
    }

}
```

Csnble

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

public class Csmble : Item
{
    public int uses;

    public override void Use()
    {
        base.Use();
        Debug.Log("Use Consumable");

        if(uses > 0)
        {
            uses--;
            FindObjectOfType<PlyrCtrlr>().Heal(amnt);
        }
        else
        {
            Remove();
        }
    }

    public override void Remove()
    {
        base.Remove();
        Debug.Log("Remove Consumable");
    }
```

```
}
```

Invtry

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

public class Invtry : MonoBehaviour
{
    public List<Item> items = new List<Item>();

    [SerializeField]
    int iSlot = 0;  //index of current equipped item
    [SerializeField]
    int nxtSlot = 0;
    [SerializeField]
    bool rot = false;

    public SpriteRenderer parentRend;

    public Item item1;
    public Item item2;
    public Item item3;


    public void Awake()
    {
        nxtSlot = iSlot;
    }

    public void Update()
    {
        if(rot)
        {
            Vector2 dir = Input.mousePosition -
Camera.main.WorldToScreenPoint(transform.position);
            float angle = Mathf.Atan2(dir.y, dir.x) * Mathf.Rad2Deg;
            if(transform.localScale.x != 1)
            {
                angle += 180;
            }
            transform.rotation = Quaternion.Lerp(transform.rotation,
Quaternion.AngleAxis(angle, Vector3.forward), Time.deltaTime * 10f);
        }

        transform.localScale = new Vector3(parentRend.flipX ? -1:1, 1, 1);

        if(Input.GetKeyDown(KeyCode.Z)) //equip an item
        {
            if (items.Count != 0)
            {
                EquipItem(nxtSlot);
                Debug.Log("Next Item!");
            }
            else
            {
                Debug.Log("No items in inventory!");
```

```
            }
        }

        if (Input.GetKeyDown(KeyCode.X)) //use equipped item
        {
            if(items[iSlot] != null)
            {
                items[iSlot].Use();
            }
        }

        if (Input.GetKeyDown(KeyCode.C)) //remove an item
        {
            if (items[iSlot] != null)
            {
                RemoveItem(items[iSlot]);
            }
        }
    }

    public void AddItem(Item item)
    {
        Item nwItem = Instantiate(item);
        nwItem.transform.SetParent(transform);
        nwItem.transform.localPosition = Vector3.zero;
        nwItem.transform.localRotation = Quaternion.identity;
        nwItem.transform.localScale = new Vector3(1, 1, 1);
        items.Add(nwItem);
        nwItem.gameObject.SetActive(false);

    }

    public void EquipItem(int slot)
    {
        if(items.Count != 0)
        {
            items[iSlot % items.Count].gameObject.SetActive(false);
            iSlot = slot % items.Count;
            items[iSlot].gameObject.SetActive(true);
            transform.rotation = Quaternion.Euler(0, 0, 0);
            rot = items[iSlot].itmRot;
            nxtSlot = (iSlot + 1) % items.Count;


        }
    }

    public void RemoveItem(Item item)
    {
        if (items.Count != 0)
        {
            items.Remove(item);
            item.gameObject.SetActive(false);
        }
    }

}
```

Item

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

public class Item : MonoBehaviour
{
    public string itmNme;
    public string itmDesc;
    public int itmCst;
    public bool itmRot;
    public float amnt;  //position add amount to plyr hlth value, weapon decrease amnt of
en hlh

    public enum rairty {common, uncommon, rare, epic, legendary};
    public rairty itmRarity;
    public Sprite itmSprte;
    protected Invtry inv;

    SpriteRenderer itmRend;

    private void Awake()
    {
        inv = FindObjectOfType<Invtry>();
        itmRend = GetComponent<SpriteRenderer>();
        itmRend.sprite = itmSprte;
    }


    public virtual void Use()
    {
        Debug.Log("base use");
    }

    public virtual void Remove()
    {
        inv.RemoveItem(this);
    }

}
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MelWpn :Item
{
    Animator anim;
    //public LayerMask enLyr;
    //public float rayLngth;

    public override void Use()
    {
        base.Use();
        Debug.Log("Use Melee Weapon");
        FindObjectOfType<Animator>().SetTrigger("Strike");

        //RaycastHit2D hit = Physics2D.Raycast(transform.position, transform.forward,
rayLngth, enLyr);
```

```
        //if(hit.collider != null && hit.collider.gameObject.CompareTag("Enemy"))
        //{
        //    Debug.Log("Hit " + hit.collider.name);
        //}

        //Debug.DrawRay(transform.position, Vector2.right * rayLngth);

    }

    public override void Remove()
    {
        base.Remove();
        Debug.Log("Remove Melee Weapon");
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            Debug.Log("Melee Hit Enemy");
            collision.GetComponent<EnCtrlr>().Dmg(amnt);
        }
    }
}
```

MelWpn

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

public class MelWpn :Item
{
    Animator anim;
    //public LayerMask enLyr;
    //public float rayLngth;

    public override void Use()
    {
        base.Use();
        Debug.Log("Use Melee Weapon");
        FindObjectOfType<Animator>().SetTrigger("Strike");

        //RaycastHit2D hit = Physics2D.Raycast(transform.position, transform.forward,
rayLngth, enLyr);

        //if(hit.collider != null && hit.collider.gameObject.CompareTag("Enemy"))
        //{
        //    Debug.Log("Hit " + hit.collider.name);
        //}

        //Debug.DrawRay(transform.position, Vector2.right * rayLngth);

    }

    public override void Remove()
    {
        base.Remove();
```

```
        Debug.Log("Remove Melee Weapon");
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            Debug.Log("Melee Hit Enemy");
            collision.GetComponent<EnCtrlr>().Dmg(amnt);
        }
    }
}
```

PrjctleCtrlr

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

public class PrjctleCtrlr : MonoBehaviour
{
    public float spd;
    Rigidbody2D bltRgdBdy;
    RngWpn colt;

    private void Awake()
    {
        bltRgdBdy = GetComponent<Rigidbody2D>();
        colt = FindObjectOfType<RngWpn>();
    }

    private void OnEnable()
    {
        bltRgdBdy.AddForce(transform.up * spd);
        Invoke("Disable", 4f);
    }

    // Start is called before the first frame update
    void Start()
    {

    }

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

    }

    private void Disable()
    {
        Destroy(gameObject);
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
```

```
            Debug.Log("Projecile Hit Enemy");
            collision.GetComponent<EnCtrlr>().Dmg(colt.amnt);

            //destroy bullet on hit
            Destroy(gameObject);
        }
    }
}
```

RngWpn

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

public class RngWpn : Item
{
    public GameObject prjctle;

    public override void Use()
    {
        base.Use();
        Debug.Log("Use Ranged Weapon");

        if (inv.transform.localScale.x == 1)
        {
            Instantiate(prjctle, transform.position, transform.rotation *
Quaternion.Euler(0, 0, -90));
        }

        if (inv.transform.localScale.x == -1)
        {
            Instantiate(prjctle, transform.position, transform.rotation *
Quaternion.Euler(0, 0, 90));
        }
    }

    public override void Remove()
    {
        base.Remove();
        Debug.Log("Remove Ranged Weapon");
    }
}
```