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

public class PlayerController : MonoBehaviour
{
    //UI
    public GameControl gameControl;

    //variables for health
    public int health = 10;
    public bool isDead = false;

    //movement
    public Rigidbody2D myRb;

    //variables for player horizontal movement
    public float baseSpeed = 4.0f;
    public float currSpeed = 4.0f;

    //variables for player vertical movement
    public int jumpCounter = 0;
    public int maxJumpCount = 2;
    public float baseJumpSpeed = 7.0f;
    public float currJumpSpeed = 7.0f;

    //variables for speed boost
    public float speedMultiplier = 2.0f;
    public float speedBoostCooldownTime = 5.0f;
    public float maxSpeedBoostTime = 2.0f;
    public float currSpeedBoostTime = 0.0f;
    public bool isSpeedy = false;
    public bool canUseSpeedBoost = true;

    //inventory
    public bool hasWeapon = true;
    public PlayerInventory Inventory;
    public Weapon CurrWeapon;
    public int currWeaponIndex;

```

```
41      //variables for player long range attack
42      public GameObject Projectile = null;
43      public float projectileSpeed = 5.0f;
44
45      //template for weapon to drop
46      public GameObject WeaponTemplate;
47
48      // Start is called before the first frame update
49      void Start()
50      {
51          gameControl = GameObject.Find("GameControl").GetComponent<GameControl>();
52
53          Inventory = gameObject.GetComponent<PlayerInventory>();
54          CurrWeapon = Inventory.WeaponList[0];
55          currWeaponIndex = 0;
56
57          myRb = gameObject.GetComponent<Rigidbody2D>();
58
59          currJumpSpeed = baseJumpSpeed;
60          currSpeed = baseSpeed;
61      }
62
63      // Update is called once per frame
64      void Update()
65      {
66          //check if we're dead
67          if (!isDead && health <= 0)
68          {
69              isDead = true;
70              transform.position = new Vector3(transform.position.x, transform.position.y, -1.0f);
71              myRb.velocity = new Vector2(0, baseJumpSpeed);
72              Destroy(gameObject.GetComponent<BoxCollider2D>());
73          }
74          if(transform.position.y < -12.0f)
75          {
76              gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
77              gameControl.UpdateLog("You lose!");
78              Destroy(gameObject);
79          }
80
```

```
81          if(!isDead)
82          {
83              //if our weapon runs out of durability, it breaks
84              if (hasWeapon && CurrWeapon.Durability <= 0)
85              {
86                  //state that the weapon has broken
87                  gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
88                  gameControl.UpdateLog(CurrWeapon.Name + " has broken.");
89
90                  //update inventory
91                  Inventory.WeaponList[currWeaponIndex] = null;
92
93                  //remove the weapon from our current weapon slot
94                  CurrWeapon = null;
95                  hasWeapon = false;
96              }
97
98              //check if the player wants to drop a weapon
99              if (Input.GetKeyDown(KeyCode.Q))
100             {
101                 if (hasWeapon)
102                 {
103                     //state that you dropped the weapon
104                     gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
105                     gameControl.UpdateLog("You dropped " + CurrWeapon.Name);
106
107                     //TODO: figure out how to do this w/o copying every single stat
108
109                     //copy current stats into new weapon
110                     WeaponTemplate.GetComponent<WeaponObject>().Name = CurrWeapon.Name;
111                     WeaponTemplate.GetComponent<WeaponObject>().MaxDurability = CurrWeapon.MaxDurability;
112                     WeaponTemplate.GetComponent<WeaponObject>().Durability = CurrWeapon.Durability;
113                     WeaponTemplate.GetComponent<WeaponObject>().Damage = CurrWeapon.Damage;
114                     WeaponTemplate.GetComponent<WeaponObject>().IsLongRange = CurrWeapon.IsLongRange;
115
116                     //drop the new weapon behind us
117                     Instantiate(WeaponTemplate, new Vector2(transform.position.x - 1.0f, transform.position.y), transform.rotation);
118
```

```
            //update inventory
            CurrWeapon = null;
            Inventory.WeaponList[currWeaponIndex] = null;
            hasWeapon = false;
        }
        else
        {
            gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
            gameControl.UpdateLog("You are not holding anything.");
        }
    }

    //check if the player wants to switch weapons
    if (Input.GetKeyDown(KeyCode.E))
    {
        //update inventory
        Inventory.WeaponList[currWeaponIndex] = CurrWeapon;
        bool switchSuccessful = Inventory.SwitchWeapons(currWeaponIndex);
        if (switchSuccessful)
        {
            gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
            gameControl.UpdateLog("You are now holding " + CurrWeapon.Name);
        }
        else
        {
            gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
            gameControl.UpdateLog("Unable to switch weapons.");
        }
    }

    //check if the player wants to move
    Move();

    //check if the player wants to attack
    if (hasWeapon)
    {
        Attack();
    }

```

```
158             //countdown for speed boost
159             if (!canUseSpeedBoost)...
177         }
178     }
179
180     void Attack()
181     {
182         //long range attack with mouse clicking
183         if (CurrWeapon.IsLongRange)
184         {
185             if (Input.GetMouseButtonDown(0))
186             {
187                 //get direction for projectile to travel
188                 Vector2 targetPos = Camera.main.ScreenToWorldPoint(new Vector2(Input.mousePosition.x, Input.mousePosition.y));
189                 Vector2 currPos = new Vector2(transform.position.x, transform.position.y);
190                 Vector2 direction = targetPos - currPos;
191                 direction.Normalize();
192
193                 Quaternion rotation = Quaternion.Euler(0, 0, Mathf.Atan2(direction.y, direction.x) * Mathf.Rad2Deg);
194
195                 //create the projectile and instantiate w/ direction and damage
196                 GameObject temp = (GameObject)Instantiate(Projectile, currPos + direction, rotation);
197                 temp.GetComponent<Projectile>().damage = CurrWeapon.Damage;
198                 temp.GetComponent<Projectile>().range = 10.0f;
199                 temp.GetComponent<Rigidbody2D>().velocity = direction * projectileSpeed;
200
201                 //lower durability
202                 CurrWeapon.Durability -= 1;
203             }
204         }
205     }
206
```

```
207     void Move()
208     {
209         //get current velocity
210         Vector2 velocity = myRb.velocity;
211
212         //check if player gets a speed boost (if they aren't already accelerating)
213         if (Input.GetKey(KeyCode.LeftShift) && !isSpeedy && canUseSpeedBoost)
214         {
215             isSpeedy = true;
216             canUseSpeedBoost = false;
217             currSpeed *= speedMultiplier;
218             currJumpSpeed *= speedMultiplier;
219         }
220
221         //check if player wants to jump
222         if (Input.GetKeyDown(KeyCode.W) && jumpCounter < maxJumpCount)
223         {
224             velocity.y = currJumpSpeed;
225             jumpCounter++;
226         }
227         //reset jump counter when we stop jumping and hit the ground
228         else if (myRb.velocity.y == 0 && myRb.IsTouchingLayers())
229         {
230             jumpCounter = 0;
231         }
232
233         //horizontal movement
234         if (Input.GetAxis("Horizontal") > 0)
235         {
236             velocity.x = currSpeed;
237         }
238         if (Input.GetAxis("Horizontal") < 0)
239         {
240             velocity.x = -currSpeed;
241         }
242
243         //move player
244         myRb.velocity = velocity;
245     }
246
```

```
247     private void OnCollisionEnter2D(Collision2D collision)
248     {
249         //check if we can pick up a weapon
250         if (collision.gameObject.transform.tag == "Weapon")
251         {
252             WeaponObject temp = collision.gameObject.GetComponent<WeaponObject>();
253             bool canPickUp = Inventory.AddWeapon(new Weapon(temp.Name, temp.MaxDurability, temp.Durability, temp.Damage, temp.IsLongRange));
254             if (canPickUp)
255             {
256                 gameControl.currTimeUntilTextDisappears = gameControl.maxTimeUntilTextDisappears;
257                 gameControl.UpdateLog("You picked up " + temp.Name);
258                 Destroy(collision.gameObject);
259             }
260         }
261     }
262 }
263
```

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

public class PlayerInventory : MonoBehaviour
{
    public Weapon[] WeaponList = new Weapon[10];

    void Awake()
    {
        //start with a basic weapon
        WeaponList[0] = new Weapon("Starting Weapon", 5, 5, 1, true);
    }

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

    }

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

    }

```

```
27      //switch to the next weapon in inventory
28      public bool SwitchWeapons(int weaponIndex)
29      {
30          int startingIndex = weaponIndex;
31          weaponIndex = (weaponIndex + 1) % (WeaponList.Length);
32
33          //keep scrolling through inventory until we get the next one
34          while(WeaponList[weaponIndex] == null)
35          {
36              weaponIndex = (weaponIndex + 1) % WeaponList.Length;
37
38              //if we're back to where we started, exit out of the loop
39              if (weaponIndex == startingIndex)
40              {
41                  return false;
42              }
43          }
44
45          //set the next weapon in inventory to the current weapon (if possible)
46          gameObject.GetComponent<PlayerController>().CurrWeapon = WeaponList[weaponIndex];
47          gameObject.GetComponent<PlayerController>().currWeaponIndex = weaponIndex;
48          gameObject.GetComponent<PlayerController>().hasWeapon = true;
49
50          return true;
51      }
52
```

```
    //add the weapon to the next available slot and return true if we had space (false if no space)
    public bool AddWeapon(Weapon newWeapon)
    {
        //if our "hand" is empty, fill that first
        if(WeaponList[gameObject.GetComponent<PlayerController>().currWeaponIndex] == null)
        {
            WeaponList[gameObject.GetComponent<PlayerController>().currWeaponIndex] = newWeapon;
            gameObject.GetComponent<PlayerController>().CurrWeapon = newWeapon;
            gameObject.GetComponent<PlayerController>().hasWeapon = true;

            return true;
        }

        //otherwise, fill the first available slot in inventory
        for(int i = 0; i < WeaponList.Length; i++)
        {
            if(WeaponList[i] == null)
            {
                WeaponList[i] = newWeapon;
                return true;
            }
        }

        //if there is no empty space, we cannot get the weapon
        return false;
    }
}
```

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

public class EnemyController : MonoBehaviour
{
    public int health = 3;
    public int damage = 1;
    public bool isFlying;
    public int score;

    private float distanceTravelled = 0.0f;
    public float minDistToShoot = 5.0f;
    private int direction = 1;

    public GameObject Projectile;
    public float currTimeToShoot = 0.0f;
    public float maxTimeToShoot = 2.0f;
    public bool canShoot = true;

    public GameObject Camera;
    public GameObject Player;
    public GameObject Coin;


    // Start is called before the first frame update
    void Start()
    {
        Camera = GameObject.Find("Main Camera");
        Player = GameObject.Find("Player");
    }

```

```
33      // Update is called once per frame
34      void Update()
35      {
36          //destroy if the enemy gets killed
37          if(health <= 0)
38          {
39              if(isFlying)
40              {
41                  score = 50;
42              }
43              else
44              {
45                  score = 10;
46              }
47              GameObject temp = Instantiate(Coin, transform.position, transform.rotation);
48              temp.GetComponent<Coin>().score = score;
49              Destroy(gameObject);
50          }
51
52          //enemy movement
53          if(Player != null && Vector2.Distance(transform.position, Player.transform.position) < minDistToShoot)
54          {
55              gameObject.GetComponent<Rigidbody2D>().velocity = Vector2.zero;
56              ShootAtPlayer(Player.transform.position);
57          }
58          else
59          {
60              if (isFlying)
61              {
62                  MoveBackAndForth(4.0f, 2.0f);
63              }
64              else
65              {
66                  MoveBackAndForth(1.0f, 1.0f);
67              }
68          }
69
```

```
        //shooting cooldown
        if(!canShoot)
        {
            currTimeToShoot += Time.deltaTime;
            if(currTimeToShoot > maxTimeToShoot)
            {
                currTimeToShoot = 0.0f;
                canShoot = true;
            }
        }
    }

    void MoveBackAndForth(float maxPacingDistance, float speed)
    {
        Vector2 newPos = transform.position;
        if (distanceTravelled < maxPacingDistance)
        {
            newPos.x += speed * direction * Time.deltaTime;
            distanceTravelled += speed * Time.deltaTime;
        }
        else
        {
            distanceTravelled = 0.0f;
            direction *= -1;
        }
        transform.position = newPos;
    }

    void ShootAtPlayer(Vector2 playerPos)
    {
        if (canShoot)
        {
            //get direction for projectile to travel
            Vector2 currPos = new Vector2(transform.position.x, transform.position.y);
            Vector2 direction = playerPos - currPos;
            direction.Normalize();

            Quaternion rotation = Quaternion.Euler(0, 0, Mathf.Atan2(direction.y, direction.x) * Mathf.Rad2Deg);

```

```
109             //create the projectile and instantiate w/ direction and damage
110             GameObject temp = (GameObject)Instantiate(Projectile, currPos + (direction * 0.75f), rotation);
111             temp.GetComponent<Projectile>().damage = damage;
112             temp.GetComponent<Projectile>().range = 5.0f;
113             temp.GetComponent<SpriteRenderer>().color = Color.black;
114             temp.GetComponent<Rigidbody2D>().velocity = direction * 5.0f;
115             canShoot = false;
116         }
117     }
118
119     private void OnBecameInvisible()
120     {
121         if (transform.position.x < Camera.transform.position.x)
122         {
123             Destroy(gameObject);
124         }
125     }
126 }
127
```

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

public class LevelGeneration : MonoBehaviour
{
    //three tiers of platforms
    public float[] groundLevels = { -4.5f, -2.5f, 0.5f };
    public Vector2 currPosToGenerate = new Vector2(9.5f, -4.5f);

    //how many tiles + where to generate
    public int minTilesToGenerate = 2;
    public int maxTilesToGenerate = 7;

    public int minBuffer = 2;
    public int maxBuffer = 3;

    //ground tiles
    public Tile GroundTile;
    public Tilemap Ground;

    //for generating grounded enemies
    public GameObject Enemy;
    public bool generateGroundedEnemy = false;
    int tileToGenerateEnemy = 0;

    //for generating flying enemies
    public float[] yPositions = { 2.0f, 2.5f, 3.0f };
    public Vector2 startPos = Vector2.zero;
    public float distTravelled = 0.0f;
    public float maxDistToGenerate = 15.0f;

    //for generating weapons
    public GameObject Weapon;

    //for calculating distance travelled + location
    public GameObject Camera;

```

```
40      // Start is called before the first frame update
41      void Start()
42      {
43          Camera = GameObject.Find("Main Camera");
44          startPos = Camera.transform.position;
45      }
46
47      // Update is called once per frame
48      void Update()
49      {
50          //if the camera is close to the "end" of the platform, generate more platforms and enemies
51          if(currPosToGenerate.x - Camera.transform.position.x <= 10.0f)
52          {
53              GenerateOnGround();
54          }
55
56          //generate flying enemies every __ m.
57          distTravelled = Vector2.Distance(Camera.transform.position, startPos);
58          if (distTravelled > maxDistToGenerate)
59          {
60              GenerateInAir();
61          }
62      }
63
```

```
64      //create objects on the ground
65      void GenerateOnGround()
66      {
67          //figure out how many tiles to generate and where to generate them
68          int numTiles = Random.Range(minTilesToGenerate, maxTilesToGenerate);
69          currPosToGenerate.y = groundLevels[Random.Range(0, 3)];
70
71          //check if we can generate a grounded enemy on the tiles
72          if (Random.Range(0, 5) > 1)
73          {
74              generateGroundedEnemy = false;
75          }
76          else
77          {
78              generateGroundedEnemy = true;
79              tileToGenerateEnemy = Random.Range(0, numTiles - 1);
80          }
81
82          //create the tiles
83          for (int i = 0; i < numTiles; i++)
84          {
85              //generate the ground
86              Vector3Int currentCell = Ground.WorldToCell(currPosToGenerate);
87              Ground.SetTile(currentCell, GroundTile);
88
89              //instantiate an enemy on the ground
90              if (generateGroundedEnemy)
91              {
92                  if (i == tileToGenerateEnemy)
93                  {
94                      GameObject temp = Instantiate(Enemy, new Vector2(currPosToGenerate.x, currPosToGenerate.y + 0.5f), transform.rotation);
95                      temp.GetComponent<EnemyController>().isFlying = false;
96                  }
97              }
```

```
98              //generate a weapon on the ground if we havent already added an enemy
99              else if (Random.Range(0, 10) < 1)
100             {
101                 GameObject temp = Instantiate(Weapon, new Vector2(currPosToGenerate.x, currPosToGenerate.y + 0.25f), transform.rotation);
102                 temp.GetComponent<WeaponObject>().Name = "Long Range Weapon";
103                 temp.GetComponent<WeaponObject>().IsLongRange = true;
104                 temp.GetComponent<WeaponObject>().MaxDurability = Random.Range(3, 6);
105                 temp.GetComponent<WeaponObject>().Durability = temp.GetComponent<WeaponObject>().MaxDurability;
106                 temp.GetComponent<WeaponObject>().Damage = 1;
107             }
108
109             //increment position
110             currPosToGenerate.x++;
111         }
112
113         //buffer before next platform
114         currPosToGenerate.x += Random.Range(minBuffer, maxBuffer);
115     }
116
117     //create objects in the air
118     void GenerateInAir()
119     {
120         GameObject temp = Instantiate(Enemy, new Vector2(Camera.transform.position.x + Random.Range(10.0f, 15.0f), yPositions[Random.Range(0, 3)]), transform.rotation);
121         temp.GetComponent<EnemyController>().isFlying = true;
122
123         distTravelled = 0.0f;
124         startPos = Camera.transform.position;
125     }
126 }
127
```

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

public class CameraScroll : MonoBehaviour
{
    public float scrollSpeed = 0.75f;
    public float speedMultiplier = 0.001f;
    public float maxSpeed = 2.0f;
    Vector3 newPos;

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

    }

    // Update is called once per frame
    void Update()
    {
        newPos = transform.position;
        newPos.x += scrollSpeed * Time.deltaTime;
        transform.position = newPos;

        if(scrollSpeed < maxSpeed)
        {
            scrollSpeed += speedMultiplier;
        }
    }
}

```

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

public class GameControl : MonoBehaviour
{
    public TextMesh Log;
    public TextMesh Score;
    public TextMesh Health;

    public float maxTimeUntilTextDisappears = 5.0f;
    public float currTimeUntilTextDisappears = 5.0f;

    public int score = 0;

    public Vector2 startPos = Vector2.zero;
    public float scoreTimer;

    public GameObject Camera;
    public GameObject Player;

    // Start is called before the first frame update
    void Start()
    {
        currTimeUntilTextDisappears = maxTimeUntilTextDisappears;

        Camera = GameObject.Find("Main Camera");

        Log = Camera.transform.Find("Log").GetComponent<TextMesh>(); ;
        Score = Camera.transform.Find("Score").GetComponent<TextMesh>();
        Health = Camera.transform.Find("Health").GetComponent<TextMesh>();

        Player = GameObject.Find("Player");
        startPos = Player.transform.position;
    }

```

```
37      // Update is called once per frame
38      void Update()
39      {
40          //run the game while the player isn't dead
41          if (Player != null)
42          {
43              if (Log.text != "")
44              {
45                  currTimeUntilTextDisappears -= Time.deltaTime;
46                  if (currTimeUntilTextDisappears <= 0)
47                  {
48                      Log.text = "";
49                      currTimeUntilTextDisappears = maxTimeUntilTextDisappears;
50                  }
51              }
52
53              scoreTimer += Time.deltaTime;
54              if (scoreTimer > 0.2f)
55              {
56                  score++;
57                  Score.text = score.ToString("00000");
58                  scoreTimer = 0.0f;
59              }
60
61              Health.text = Player.GetComponent<PlayerController>().health.ToString("00");
62          }
63          //whent he player dies, stop scrolling
64          else
65          {
66              Camera.GetComponent<CameraScroll>().scrollSpeed = 0.0f;
67              Camera.GetComponent<CameraScroll>().speedMultiplier = 0.0f;
68          }
69      }
70
```

```
71      public void UpdateLog(string message)
72      {
73          if (Log != null)
74          {
75              Log.text = message;
76          }
77      }
78
79      public void UpdateScore(int addedScore)
80      {
81          score += addedScore;
82      }
83  }
84
```

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

public class Weapon
{
    public string Name;
    public int MaxDurability;
    public int Durability;
    public int Damage;
    public bool IsLongRange;

    public Weapon(string name, int maxDurability, int durability, int damage, bool isLongRange)
    {
        Name = name;
        MaxDurability = maxDurability;
        Durability = durability;
        Damage = damage;
        IsLongRange = isLongRange;
    }
}

```

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

public class WeaponObject : MonoBehaviour
{
    public string Name;
    public int MaxDurability;
    public int Durability;
    public int Damage;
    public bool IsLongRange;

    public GameObject Camera;

    void Start()
    {
        Camera = GameObject.Find("Main Camera");
    }

    private void OnBecameInvisible()
    {
        if (transform.position.x < Camera.transform.position.x)
        {
            Destroy(gameObject);
        }
    }
}
```

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

public class Projectile : MonoBehaviour
{
    public int damage;
    public float range = float.MaxValue;
    public Vector2 startPos = Vector2.zero;
    public float distTravelled = 0.0f;

    // Start is called before the first frame update
    void Start()
    {
        startPos = transform.position;
    }

    // Update is called once per frame
    void Update()
    {
        if(Vector2.Distance(startPos, transform.position) >= range)
        {
            Destroy(gameObject);
        }
    }

```

```
    private void OnCollisionEnter2D(Collision2D collision)
    {
        if(collision.transform.gameObject.tag == "Enemy")
        {
            collision.transform.gameObject.GetComponent<EnemyController>().health -= damage;
        }

        if(collision.transform.gameObject.tag == "Player")
        {
            collision.transform.gameObject.GetComponent<PlayerController>().health -= damage;
        }

        Destroy(gameObject);
    }

    private void OnBecameInvisible()
    {
        Destroy(gameObject);
    }
}
```

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

public class Coin : MonoBehaviour
{
    public int score;

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

    }

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

    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        if(collision.transform.gameObject.tag == "Player")
        {
            GameObject.Find("GameControl").GetComponent<GameControl>().UpdateScore(score);
            Destroy(gameObject);
        }
    }
}

```