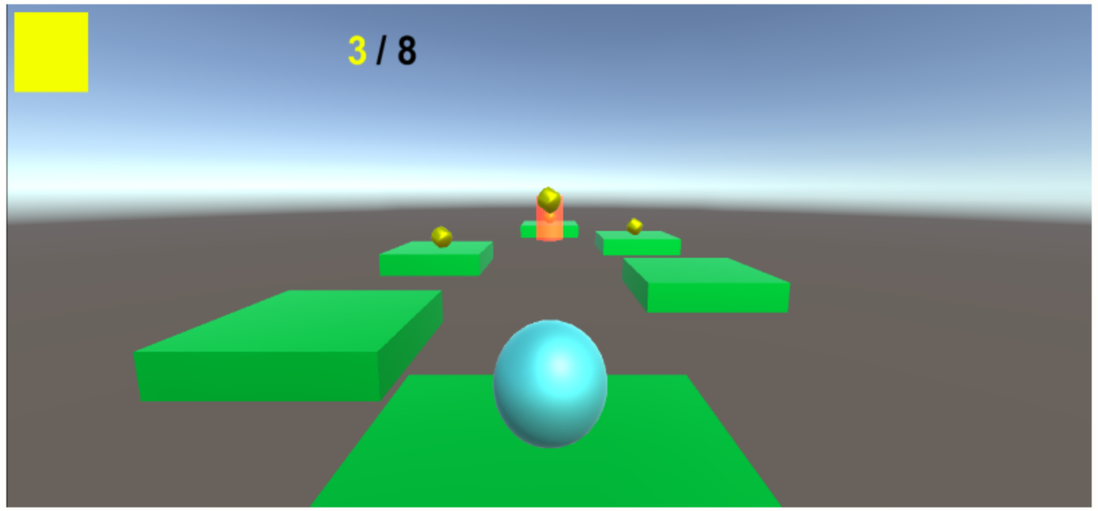
3D UNITY RIGHT PROJECT

Unity,C#을 활용한 간단한 유니티 공굴리기 게임

권용규.

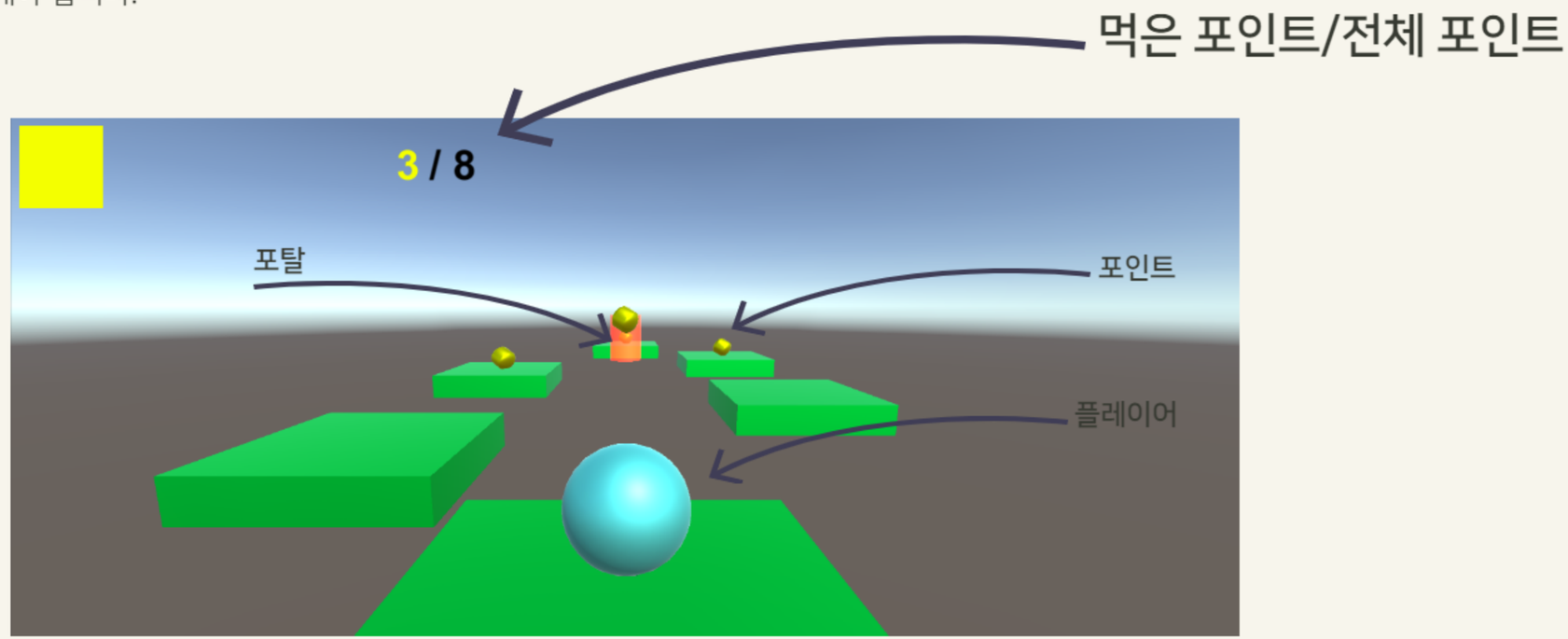


목차

기본 게임 설명

점프 공굴리기 게임

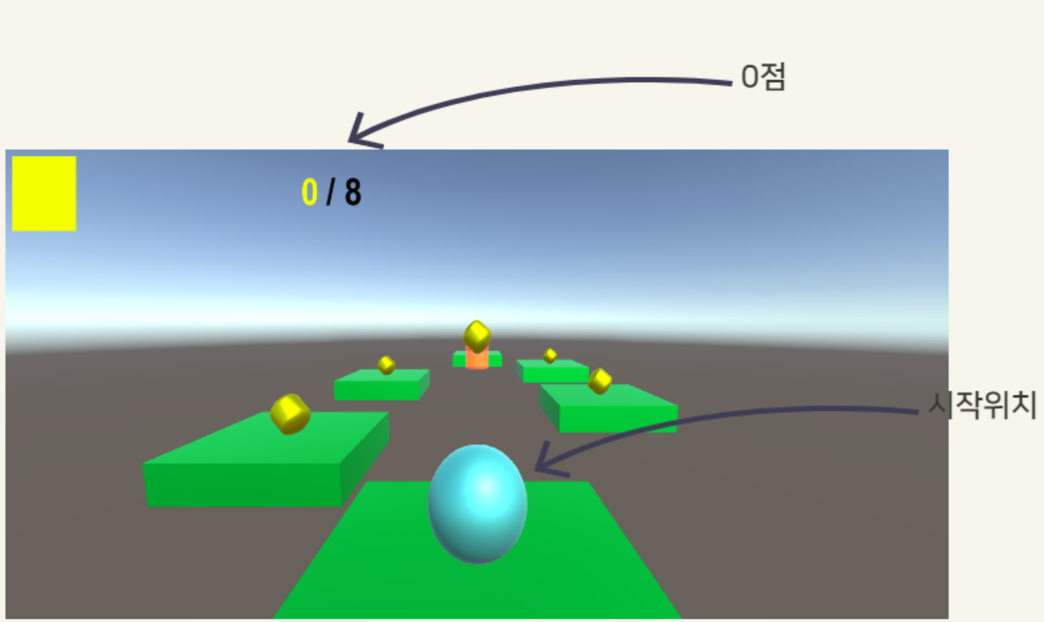
간단하게 플랫과 구,기둥을 활용한 오브젝트를 만들고 물리 효과와 조건문 로직을 사용하였습니다. 모든 점수를 얻고 포탈을 타면 다음으로 이동되고 그렇지 않을 경우 처음부터 다시 진행해야 합니다.

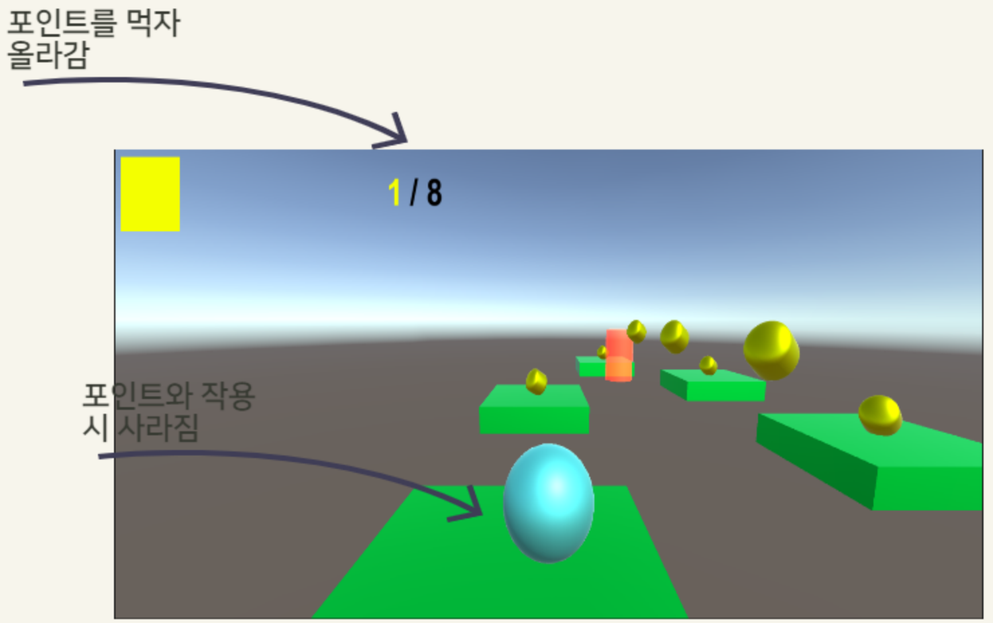


상세 설명

점프 공굴리기 게임

기본적으로 점프기능(최대 2번까지 점프)을 만들었고 포인트에 회전기능을 넣었으며 플레이어인 구체가 접촉시 사라지고 위에 UI단에서 숫자가 1 올라가는 모습을 볼수 있다.

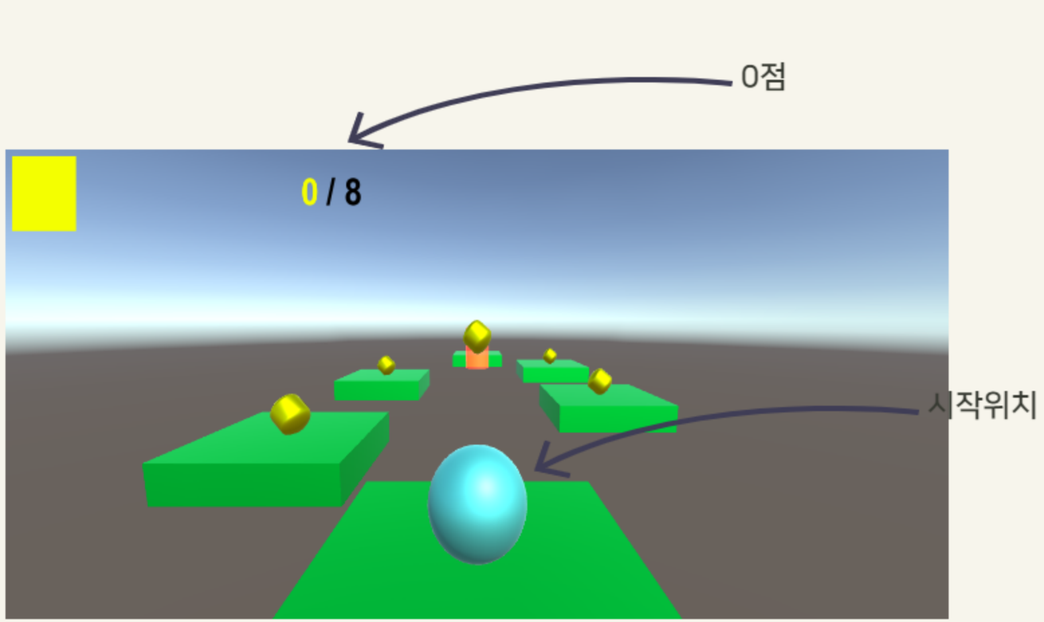


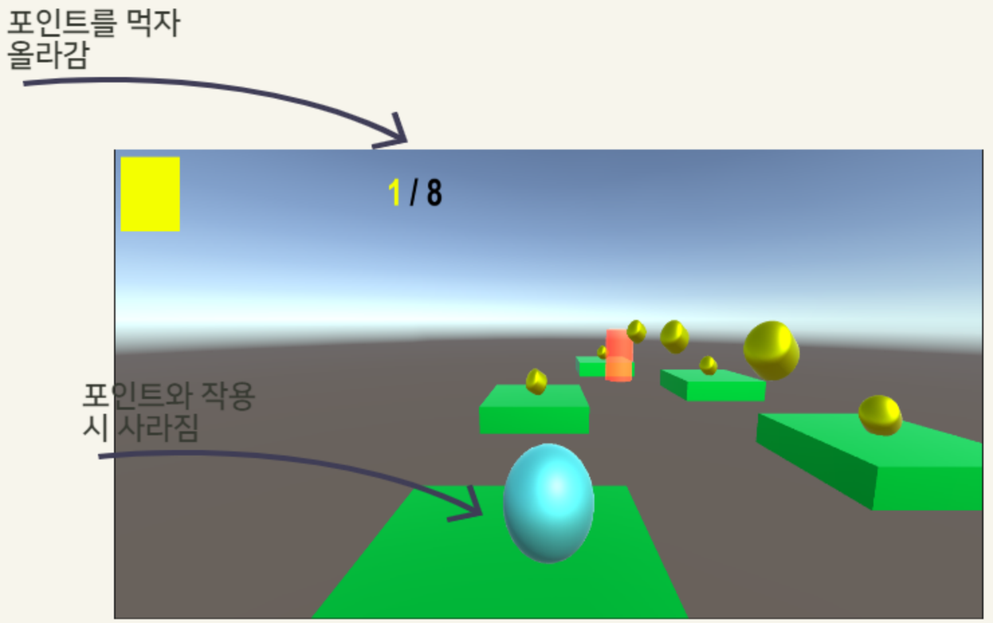


상세 설명

점프 공굴리기 게임

기본적으로 점프기능(최대 2번까지 점프)을 만들었고 포인트에 회전기능을 넣었으며 플레이어인 구체가 접촉시 사라지고 위에 UI단에서 숫자가 1 올라가는 모습을 볼수 있다.

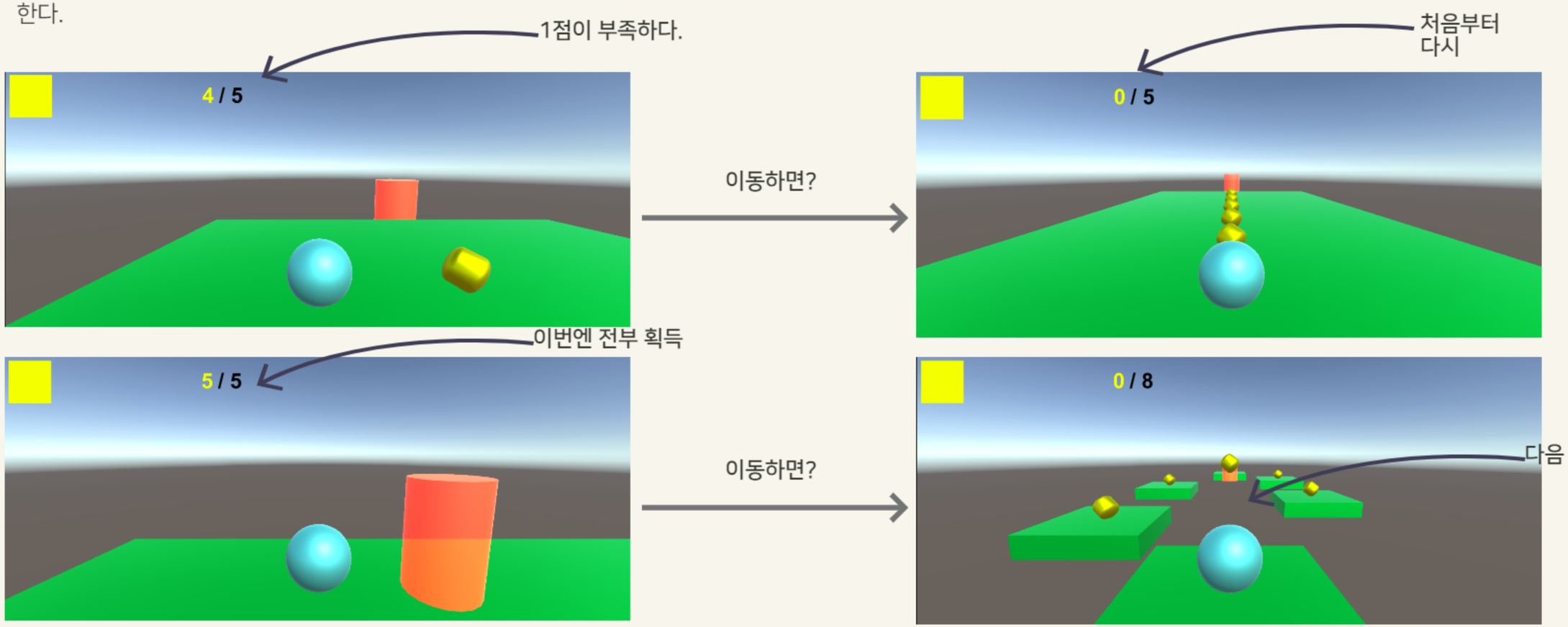




상세 설명

점프 공굴리기 게임

해당 스테이지에 존재하는 모든 점수를 획득한 후 포탈에 이동시 다음 맵으로 이동할수 있게끔 만들었다 만약 1개라도 포인트를 덜 획득할시 처음부터 다시 시작한다.



코드 설명

1.플레이어 이동

```
public class Player_Move : MonoBehaviour
{
    1 reference
    public float jumpPower;

    3 references
    public int itemCount;

    6 references
    public Game_Manager GM;

    4 references
    int count = 0;

    4 references
    bool isDoubleJump;
    3 references
    Rigidbody rigid;

    2 references
    AudioSource audio;

    0 references
    void Awake()
    {
        isDoubleJump = false;
        rigid = GetComponent<Rigidbody>();
        audio = GetComponent<AudioSource>();
    }
```

소수형 점프파워

숫자형 포인트 수

다른 스크립트 참조

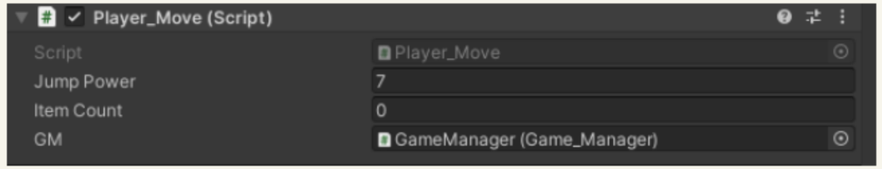
Unity 물리 관련 선언

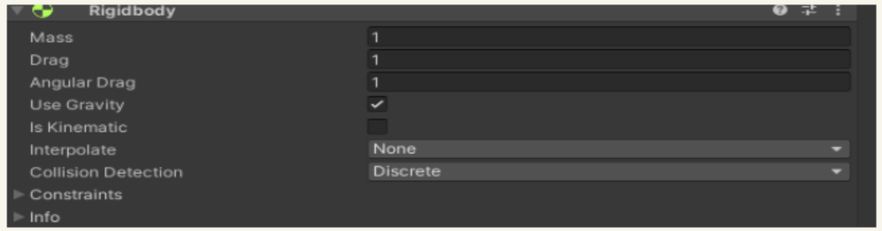
Unity 사운드 관련 선언

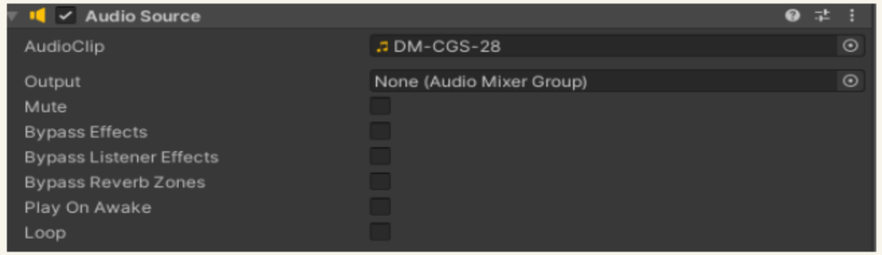
스크립트를 작성하고 오브젝트에 적용시 선언된 변수를 Unity화면에서도 확인 가능하면 임의로 변경 가능하다.

유니티에서 지원하는 옵션을 가져다가 사용하기 위해서 참조해서 임의로 변수명을 선언해준다.

Awake() 매서드 : 첫 실행시 한번만 실행되는 매서드로 Unity 에서 지원하는 옵션을 GetComponent<이름>으로 가져와 적용시켜준다.







코드 설명

1.플레이어 이동

Update() 매서드: 주기적으로 초기화 해주어야 하는것들을 담는다. 해당 코드에선 플레이어가 2단 점프를 하기 위해 조건문을 담았고 isDoubleJump의 참,거짓 여부에 따라 점프를 더하거나 점프에 제한을 걸었다.

AddForce():Add 더하다 Force 힘, 해당 조건문에서 Jump버튼 입력을 받고, 더블 점프가 거짓이라면 점프버튼(스페이스바)입력시 공이 점프한다.

OntriggerEnter() 매서드 : 다른 오브젝트와 반응 이벤트를 담는 매서드, 다른 오브젝트 중 isTrigger이 true인 오브젝트에 반응한다. 해당 코드에선 포인트와 반응시 점수가 오르는 것과 골인지점에 반응시 포인트 갯수 여부를 판별해 다시 시작할지 다음 스테이지로 넘어가는지를 담았다.

```
0 references
void Update(){

    if(Input.GetButtonDown("Jump") && !isDoubleJump){
        rigid.AddForce(new Vector3(0,jumpPower,0), ForceMode.Impulse);
        count++;
    }
    if(count > 1){
        count = 0;
        isDoubleJump = true;
    }
}
0 references
void FixedUpdate()
{
    float h = Input.GetAxisRaw("Horizontal")/2;
    float v = Input.GetAxisRaw("Vertical")/2;

    rigid.AddForce(new Vector3(h,0,v), ForceMode.Impulse);
}

0 references
void OnCollisionEnter(Collision collision) {

    if(collision.gameObject.tag == "Floor"){
        count = 0;
        isDoubleJump = false;
    }
}
```

만약 점프를 2번해서 카운트가 1보다 커질 경우 isDoubleJump를 참으로 바꾸어 조건문이 돌지 않게끔 바꾼다

FixedUpdate() 매서드:Update처럼 프레임 기반 호출이 아닌 Fixed Timestep에 설정된 값에 따라 일정한 간격으로 호출됨
여기선 Horizontal,Vertical에 해당되는 반응을 받을경우 AddForce로 오브젝트에 힘을 주어 상 하 좌 우 로 움직이게끔 하였다.

OnCollisionEnter()매서드 : 충돌 이벤트를 담는 매서드 이다
해당 코드에선 오브젝트 태그 값이 Floor 인 것과 플레이어가 닿을경우 점프 카운트를 0으로 초기화 하고 isDoubleJump값을 거짓으로 바꾸어 다시 점프를 할수있게끔 해준다.

```
void OnTriggerEnter(Collider other) {
    if(other.tag == "Point"){
        itemCount++;
        audio.Play();
        other.gameObject.SetActive(false);
        GM.GetItem(itemCount);
    }
    else if(other.tag == "Finish"){
        if(itemCount == GM.TotalItemCount){
            if(GM.stage == 3){
                SceneManager.LoadScene("stage1_0");
            }
            else
                SceneManager.LoadScene("stage1_"+(GM.stage + 1));
        }
        else{
            SceneManager.LoadScene("stage1_"+GM.stage);
        }
    }
    Debug.Log(GM.stage);
}
```

코드 설명

2.카메라 무브 및
포인트 회전

Rotate:회전
Update매서드를 사용해서 오브젝트에 Rotate 회전을 부여했다
Left를 사용하여 회전할 축을 지정하고 Time.DeltaTime을 사용하여 프레임당 회전력을 부여했고
Space.World를 사용해서 물체를 기울여도 월드 좌표를 기준으로 회전하게끔 만들었다.

```
0 references
public class Point_Event : MonoBehaviour
{
    1 reference
    public float rotateSpeed;
    0 references
    void Update()
    {
        transform.Rotate(Vector3.left * rotateSpeed * Time.deltaTime, Space.World);
    }
}
```

Awake로 게임내 태그가 player인 오브젝트의 위치를 찾는다, 그런후 자신의 위치보정을 맞춘다

LateUpdate() 매서드 : Update 관련 수행중 Update() 가 작동된 이후 작동됨, 주기적으로 카메라의 위치를 플레이어 위치 + 위에서 맞춘 위치보정을 주기적으로 초기화 시켜서 플레이어를 따라가게끔 만들었다.

```
public class Camera_Move : MonoBehaviour
{
    3 references
    Transform playerTransform;
    2 references
    Vector3 Offset;
    0 references
    void Awake()
    {
        playerTransform = GameObject.FindGameObjectWithTag("Player").transform;
        Offset = transform.position - playerTransform.position;
    }

    // Update is called once per frame
    0 references
    void LateUpdate()
    {
        transform.position = playerTransform.position + Offset;
    }
}
```

코드 설명

3.게임 매니저

using UnityEngine.SceneManagment
using UnityEngin.UI
Unity 에서 지원하는 해당 기능을 가져온다.

```
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

1 reference
public class Game_Manager : MonoBehaviour
{
    2 references
    public int TotalItemCount;
    6 references
    public int stage;
    1 reference
    public Text stageCount;
    public Text playerCount;
    1 reference
    void Awake()
    0 references
    {
        stageCount.text = "/ " + TotalItemCount;
    }

    public void GetItem(int count)
    {
        1 reference
        playerCount.text = count.ToString();
    }
    private void OnTriggerEnter(Collider other) {
        if(other.gameObject.tag == "Player"){
            0 references
            SceneManager.LoadScene(stage);
        }
    }
}
```

Awake() 매서드를 통해 시작될 때 UI에 있는 스테이지에 스테이지 레벨을 표시해준다.

포인트를 획득할때 UI에서 포인트 갯수 증가 를 발생시켜준다.

조건문을통해 플레이어와 트리거 이벤트 발생시 장면을 로드해온다(이때 로드시 Awake() 매서드 부터 다시 실행)

프로젝트를 진행하며 느낀점.

구글링과 유튜브, Unity Doc을 보면서 차근차근 진행해보았다. 처음엔 Update(), FixedUpdate, LateUpdate()의 차이가 구별되지 않았고 전부다 비슷한 녀석이라고 생각했다.

처음 진행해본 Unity 프로젝트지만 프로그램 코딩이 아닌 게임을 코딩한다는 기분이라서 그런가 꽤나 재밌게 코딩한것 같다. 살면서 누구나 한번쯤 게임 코딩을 해보면 재밌을것 같다란 생각을 한번씩은 해볼텐데 의외로 입문 난이도도 그렇게 높지 않고 C#을 어느정도 다룰줄 안다면 Unity도 배울만한 가치가 있고 해볼만하다고 생각하게된 시간이다.

Thank You

봐주셔서 감사합니다

권용규