6장 닷지( 플레이어 제작)

머터리얼 : 유니티에서 게임 오브젝트의 컬러를 결정하는 것( 세이더와 텍스처가 합해진 에셋으로 오브젝트의 픽셀 컬러를 결정한다)

알베도( 반사율) : RGB와 투명도를 결정할수있다.

카메라 설정 : 메인 카메라에서 게임화면이 어떤식으로 보일지 설정 가능, 오브젝트 외에도 다른곳의 화면 배경설정 도한 메인 카메라에서 확인 가능

하이어라키창에서 바닥과 가로막을 벽을 구성

플레이어 생성시 태그에 플레이어 태그를 설정해야 플레이어와 다른 오브젝트들간의 충돌시 확인하기 용이함

플레이어에게 중력과 물리현상을 구현하기 위해 Rigidbody 컴포넌트 추가

Freeze Position Y와 Freeze Rotation x,y를 체크하여 위로 튀어 오르거나 앞옆으로 흔들리는 것을 방지

스크립트 작성시 start()메서드에 사용될 컴포넌트의 변수와 이용될 값들에 대한 변수 선언

스크립트가 시작될 시 필요한 값들을 불러온다

사용자 입력 감지

유니티에서 update()메서드는 한 프레임당 한번 실행되는 메서드이다. 60프레임을 지원하는 프로그램과 모니터가 있다면 1초에 60번 새로운 화면을 그리는 프로그램이 생성이 되는것이다.

Update() 메서드에 키 입력과 이동 방향 넣기



키코드는 아스키 코드처럼 대응대는 숫자가 있지만 다 외울 수 없으므로 대응되는 변수를 만들어 놓았다.

Input.GetKey()의 경우 키를 누르는 동안 true, 그 외에는 false반환

Input.GetKeyDown(): 키를 누르는 순간 true, 그 외에는 false

Input.GetKeyUp() 키를 눌렀다 때는 순간 true, 그 외에는 false

Die() 메서드 추가

Die 메서드를 만들지만 스스로 사용하는 것이 아닌 충돌한 객체가 playerController 컴퍼넌트에 접근하여 실행

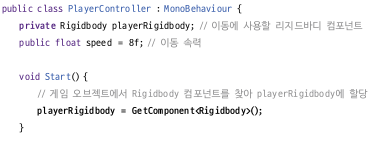
 다음과 같이 선언

gameObject : 컴포넌트 입장에서 자신이 추가도니 게임 오브젝트를 가리키는 변수. GameObject 타입의 변수이며 MonoBehaviour에서 제공 컴포넌트 들은 gameobject를 사용하여 자신을 사용중인 게임 오브젝트에 접근 가능

setActive()를 사용해서 자신을 비활성화 : 씬에서 안보이게 됨

playerController 개선하기

1. 조작이 게임에 즉시 반영 X
2. 입력 감지코드가 복잡
3. 컴포넌트를 드래그&드롭이 아닌 변수로 선언해보기



Private 으로 선언하여 더 이상 인스펙터 창에서 보이지 않는다 , GetComponet<타입이름>()으로 컴포넌트를 불러온다.



//점프기능 구현해보기

playerRigidbody.velocity = newVelocity;

if (Input.GetKey(KeyCode.Space))

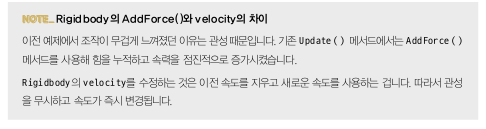
playerRigidbody.velocity = new Vector3(0f, 3f, 0f);

**해당 오브젝트에 Constant Force 컴포넌트를 추가 하고 Force 벡터 값에 원하는 중력값을 넣습니다.**

Input.GetKey()가 아닌 input.GetAxis()로 키 값이 아니라 축에 대한 입력값을 받는다.



Velocity 변수는 속도를 표현할수 있다



입력 매니저

움직임에 직접 키를 할당 하는 것이 아니라 유니티에서 지원하는 입력 매니저를 설정하면 멀티 플랫폼을 위해 지원할때도 다른키로 변환할때도 편리하게 변경할수있다.

키보드로 할당으로 할시 bool값을 받지만 숫자로 받을시 아날로그스틱처럼 정도 조절이 가능하다

Edit -> project setting -> input manager 로 관리