 **Component**

**Collider**

[설명]

콜라이더 컴포넌트는 물리 충돌 처리를 위해서 사용하는 컴포넌트다. 보이지 않는 요소이므로 오브젝트의 메시와 동일한 모양일 필요는 없으며 일반적으로 3D에서는 BoxCollider, SphereCollider, CapsuleCollider 등을 사용하며 2D에서는 BoxCollider2D, CircleCollider2D 등을 사용한다. 이 외에도 EdgeCollider2D, PolygonCollider2D, TilemapCollider2D, MeshCollider 등 여러 종류의 콜라이더 컴포넌트가 존재한다.

[필드/메소드]

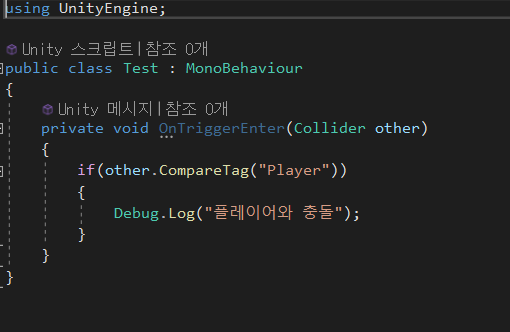
콜라이더를 가진 충돌체끼리는 서로 영역이 겹치면 충돌이 발생하며 IsTrigger 설정이 체크되어있을 경우 충돌 시 부딪히지 않고 겹침 상태만 검사한다. 충돌과 관련해서 여러 메소드가 호출된다..

|  |  |
| --- | --- |
| OnCollisionEnter(Collision) | 충돌이 발생하는 순간 |
| OnCollisionStay(Collision) | 충돌이 유지되는 동안 |
| OnCollisionExit(Collision) | 충돌이 끝나는 순간 |

IsTrigger 설정이 체크되어 있는 오브젝트와의 충돌에서는 OnTriggerEnter(Collision)처럼 Trigger라고 쓴다.

2D의 경우, OnTriggerEnter2D(Collider2D)처럼 2D가 뒤에 붙는다.

[예제]



해당 코드는 Collider 컴포넌트와 Test스크립트가 붙은 오브젝트가 다른 충돌체와 충돌했을 때(둘 중 하나는 IsTrigger가 체크되어있다) 충돌한 오브젝트의 태그(Tag)가 Player라면 밑의 로그를 콘솔창에 표시한다.

**Component**

**Rigidbody**

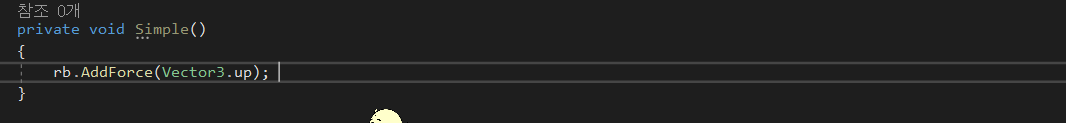
[설명]

GameObject가 물리 제어로 동작하게 해준다. 2D에서는 Rigidbody2D를 사용하며 힘과 토크를 받아 오브젝트가 사실적으로 움직이도록 해준다. 이 컴포넌트를 사용하면 스크립팅을 통해 가해진 힘으로 움직이거나 중력의 영향을 받게 할 수 있다.

[필드/메소드]

|  |  |
| --- | --- |
| Mass | 오브젝트의 질량이다 |
| Drag (2D에서는 Linear Drag) | 오브젝트가 힘에 의해 움직일 때 공기저항이 미치는 정도. 0이면 공기저항이 없다. |
| Angular Drag | 오브젝트가 회전운동을 할 때의 공기 저항이 미치는 정도이다. |
| Use Gravity (3D) | 활성화되면 오브젝트가 중력의 영향을 받는다 |
| Gravity Scale (2D) | 오브젝트가 중력에 의해 영향을 받는 정도 |
| Is Kinematic (3D) | 활성화되면 오브젝트가 물리엔진으로 제어되지 않고 오로지 Transform으로만 조작된다. |
| Constraints | 움직임에 대한 제약사항을 설정한다. 활용하면 원하는 축으로 움직이는 것이나 회전하는 것을 막을 수 있다. |
| AddForce(Vector3 force, Forcemode mode) | 오브젝트에 힘을 준다. |
| velocity | 속력 벡터를 나타내며 인스펙터에서 수정할 수 없다. |

[예제]



해당 예제는 Rigidbody가 붙은 오브젝트한테 위 방향으로 (0,1,0)힘을 줘서 움직이게 하는 코드이다.

**Component**

**SpriteRenderer**

[설명]

Sprite를 렌더링하고 스프라이트가 2D 및 3D 프로젝트의 씬에 시각적으로 표시되는 방식을 제어한다.

2D 프로젝트에서 많이 쓰이는 컴포넌트로 하이라키(Hierarchy)에서도 바로 생성할 수 있다.

[필드/메소드]

|  |  |
| --- | --- |
| Sprite | 렌더링할 스프라이트 텍스처를 정의한다. |
| Color | 스프라이트의 색을 정할 수 있다. 이미지에 색조를 더하거나 다른 컬러를 적용할 수 있다. |
| Flip | 선택한 축에 있는 스프라이트 텍스처를 뒤집는다. |
| Material | 스프라이트를 표시할 때 사용하는 재질을 설정한다. 스프라이트에 특수한 효과를 추가할 때 많이 사용한다. |
| Sorting Layer | 렌더링 중 우선 순위를 제어하는 Sprite의 정렬 레이어를 설정한다. 드롭 다운 상자에서 기존 레이어를 선택하거나 새 레이어를 만들 수 있으며 밑에 있을수록 우선순위가 높다. |
| Order In Layer | 소팅 레이어로 지정된 순서를 더욱 자세하게 지정하는 우선순위로 번호가 높을수록 앞에 보이게 된다. |
| Mask Interaction | Sprite Mask와 상호작용할 때 취하는 동장을 설정한다.  Visible Inside Mask : 마스크가 오버레이하는 곳에 스프라이트가 표시됨.  Visible Outside Mask: 마스크가 오버레이하는 곳 말고 밖에서 스프라이트가 보인다. |

[추가내용]

Sprite Mask 컴포넌트로 스프라이트 또는 스프라이트 그룹을 숨길 수 있는데 Sprite Renderer컴포넌트를 사용하는 오브젝트에만 영향을 미친다.

**Component**

**MeshRenderer**

[설명]

.MeshFilter에서 정해진 메쉬를 그려준다..

MeshFilter : 에셋에서 메쉬를 가져와 화면상에서 렌더링하기 위해 MeshRenderer에 전달한다.

Mesh : 입자 크기를 나타내는 단위로 쉽게 말해서 입체 도형이라고 보면 된다.

[필드/메소드]

|  |  |
| --- | --- |
| Materials의 Size와 Element | Material의 개수를 정하고 Element를 통해서 Material파일을 정해줄 수 있다. |
| CastShadows | 그림자를 켜고 끄는 기능 |
| Receive Shadows | 다른 객체의 그림자에 영향을 받게 할지 정함. |
| Contribute Global Illumination | 다른 객체가 반사한 빛과 간접 광등이 나타날지 정함 |

[추가내용]

Skinned Mesh Renderer은 메쉬의 형태가 애니메이션(Bone)에 따라 움직이는 오브젝트를 렌더링하기 위한 컴포넌트이다. Skinned Mesh Renderer은 에셋에 임포트할 때 이를 필요로 하는 모든 Mesh에 자동으로 추가된다.

**Component**

**AudioSource**

[설명]

할당된 AudioClip의 소리를 발생시키는 컴포넌트이다.

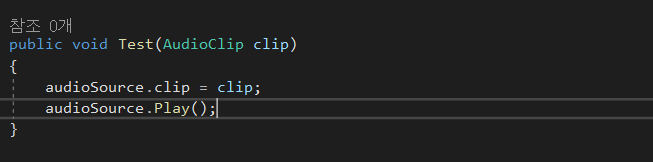
AudioClip : 오디오 데이터를 저장하는 공간 (오디오 파일)

AudioListener : 마이크와 같은 장치로 Audio Source로부터 받은 사운드를 재생하는 역할을 하며 프로젝트 생성 시에 Main Camera에 추가되어 있다.

[필드/메소드]

|  |  |
| --- | --- |
| AudioClip | 재생될 사운드 클립 |
| Output | 기본적으로 클립은 오디오 리스너에 직접 출력되지만 이 프로퍼티를 사용하면 오디오 믹서에 출력한다. |
| Mute | 음소거 상태로 할지 정함 |
| Play On Awake | 활성화하면 씬이 실행되는 시점에 사운드 재생이 시작된다. 스크립트에서 Play함수를 통해서 실행할 수도 있다. |
| Loop | 오디오 재생이 끝나면 처음부터 다시 오디오를 재생함 |
| Volume | 소리의 크기 |
| Pitch | 소리의 음높이 |

[간단 예제]



다음은 매개변수로 AudioClip을 받아서 AudioSource의 AudioClip에 할당하고 해당 사운드를 재생시키는 간단한 코드이다.

**Component**

**ParticleSystem**

[설명]

작고 간단한 이미지나 메쉬를 대량으로 표현하는데 사용하도록 만들어진 시스템이다. 크기, 방향, 모양 등을 설정하여 눈이나 비 등의 자연환경 효과, 게임 속의 스킬 이펙트 등 여러 곳에 활용이 된다..

[필드/메소드]

|  |  |
| --- | --- |
| Duration | 파티클 시스템의 재생시간 |
| Looping | 반복사용 설정이다. |
| Prewarm | Looping이 선택되면 활성화된다. 한 사이클(One Duration)이 이미 완료된 상태에서 파티클 시스템이 동작한다. |
| Start Delay | 시작 시간을 연기한다 |
| Start Lifetime | 파티클의 재생시간 (시스템이 아님) |
| Start Speed | 파티클의 시작 속도 |
| Start Size | 파티클의 시작 크기 |
| Start Rotation | 파티클의 시작 시 회전 값 설정 |
| Start Color | 파티클 시작 색상 |
| Gravity Modifier | 파티클에 중력에 영향 받을지 설정 |
| Delta Time | Unscaled와 Scaled값을 설정함.. Unscaled로 하면 Time.timeScale에 영향을 받지 않는다. |

[추가 내용]

파티클 시스템에는 여러 모듈(Module)이 있고 그 모듈에도 각각의 속성들이 있어서 파티클을 더 세밀하게 설정할 수 있다. 많이 쓰는 모듈에는 다음과 같은 것들이 있다.

|  |  |
| --- | --- |
| Emission | 방출되는 입자 수를 제어한다 |
| Shape | 파티클을 내보내는 모양 설정 |
| ~ Over Lifetime | 파티클이 활성화된 시간 동안에 어떤 속성을 조절함 |
| Collision | 파티클과 게임오브젝트가 충돌하도록 설정함 |
| Renderer | 렌더러 설정에 따라 파티클의 이미지나 메시가 렌더링 되는지를 결정함 |

**Script**

**생명 주기(Life Cycle)**

[설명]

유니티 스크립트의 대부분은 MonoBehaviour클래스를 상속받아 만들어지는데 이 클래스를 상속받은 오브젝트는 Scene에 있는 동안 엔진에서 일정한 흐름에 따라 자동으로 함수를 호출하는데, 이 패턴의 흐름이 생명 주기이다.

|  |  |
| --- | --- |
| Awake | Start함수의 이전 및 프리팹의 인스턴스화 직후에 호출된다. 게임을 실행할 때 게임오브젝트가 비활성화 상태라면 오브젝트가 활성화되면 Awake함수가 호출된다 |
| OnEnable | 오브젝트가 활성화 한 직후에 호출된다. 게임오브젝트를 껐다/켰다 반복하면 여러 번 호출된다. |
| Start | 스크립트의 인스턴스가 활성화되면 첫 번째 프레임의 업데이트 전에 호출된다. |
| Update | 매 프레임마다 한 번 씩 호출되며 일반적으로 1초당 60프레임이며 이는 1초에 60번 Update함수가 실행되는 것이다. |
| FixedUpdate | Fixed Timestep에 설정된 값에 따라 일정한 간격으로 호출된다. 물리 효과가 적용된 오브젝트를 조정할 때 사용한다. (Update는 불규칙한 호출이라서 물리엔진 충돌검사 등이 제대로 안될 수 있다.) |
| LateUpdate | Update 후 프레임마다 한 번씩 호출된다. Update에서 수행되는 계산이 완료되면 LateUpdate함수가 시작된다. |
| OnDisable | 오브젝트가 비활성화 될 때 호출한다. (OnEnable과 마찬가지로 여러 번 호출 될 수 있다.) |
| OnDestroy | 오브젝트 파괴될 때 호출된다. (Object.Destroy함수 또는 씬 종료시에 파괴된다.) |
| OnApplicationQuit | 응용 프로그램 종료 전에 모든 게임 오브젝트에서 호출한다. |

[실행 순서]

Awake -> OnEnable -> Start -> FixedUpdate -> OnTriggerXXX -> OnCollisionXXX -> Update -> LateUpdate -> OnApplicationQuit -> OnDisable -> OnDestroy

**Component**

**EventTrigger**

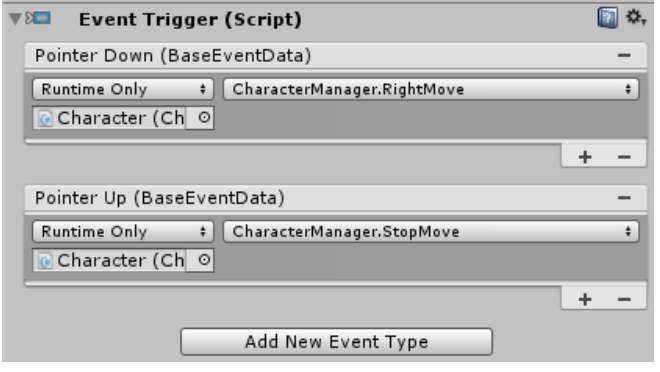
[설명]

Canvas 내에 있는 이미지들에 대한 이벤트를 받아들인다.

EventSystem이 어떤 게임 오브젝트를 대상으로 어떤 형태의 이벤트를 발동시킬 때마다 그것을 캐치해서 EventTrigger에 등록된 함수들을 실행시킴.

[자주 쓰는 이벤트]

|  |  |
| --- | --- |
| PointerDown | 개체를 눌렀을 때 호출 (클릭) |
| PointerUp | 누르고 있는 개체에서 뗐을 때 호출. |
| PointerEnter | 개체에 포인터가 들어갔을 때 |
| PointerExit | 개체에서 포인터가 나왔을 때 |
| BeginDrag | 드래그가 시작되기 전에 호출 |
| Drag | 드래그하는 동안 포인터가 이동할 때마다 호출 |
| Drop | 어떤 개체에 (포인터)드롭할 때 |
| EndDrag | 드래그가 끝날 때 호출 |



다음은 스크립트에서 PointerDown과 PointerUp이벤트를 만들고 그 때 호출될 함수를 설정해서 EventTrigger에 연결시키는 코드이다.