

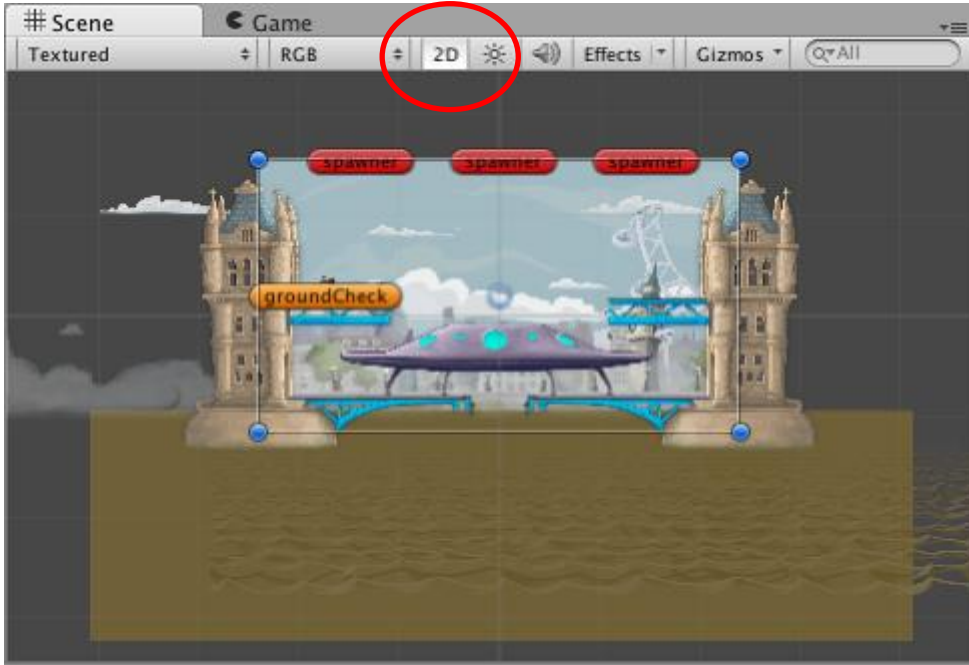
Chap2. 2D

Unity

Contents

- Sprite
 - Sprite 생성기, 에디터, 패커
 - 정렬 그룹
 - 9슬라이싱 Sprite
 - Sprite 마스크
 - Sprite 아틀라스
- 물리 레퍼런스 2D Physics Reference 2D
 - 물리 2D 설정
 - 리지드바디 2D
 - Collider 2D
 - 서클 Collider, 박스 Collider, 에지 Collider, 캡슐 Collider, 복합 Collider
 - 물리 머티리얼 2D
 - 2D 조인트
 - 거리 조인트, 고정 조인트, 마찰 조인트, 힌지 조인트, 상대 조인트, 슬라이더 조인트, 스프링 조인트, 타겟 조인트, 휠 조인트
 - Constant Force 2D
 - 이펙터 2D
 - 영역 이펙터, 부력 이펙터, 포인트 이펙터, 플랫폼 이펙터, 표면 이펙터

01. 2D에서의 게임 플레이



- 2D 버튼이 클릭되면,
 - Orthogonal View로 설정
 - 카메라는 Z축에서 Y축이 위로 향하도록 설정
 - 이 상태에서 2D 오브젝트들을 쉽게 배치할 수 있음
- 2D 그래픽스
 - Sprite : 2D Graphic Object의 기본 Texture
 - Sprite Editor : 큰 이미지에서 Sprite Graphic 추출
 - Object Rendering : 3D의 mesh renderer 대신 Sprite Renderer 컴포넌트를 사용
 - 컴포넌트 메뉴(Component > Rendering > Sprite Renderer)를 통해 게임 오브젝트에 추가하거나
 - 메뉴(GameObject > 2D Object > Sprite)를 통해 Sprite Renderer가 이미 연결된 게임 오브젝트를 즉시 생성
- 2D 물리
 - 2D 물리만 다루는 별도의 물리 엔진을 사용(최적화)
 - Rigidbody 2D, Box Collider 2D, Hinge Joint 2D등

02. Sprite

Unity에서는 2D 그래픽 오브젝트(Sprite)를 효율적으로 관리하기 위한 기법 및 도구를 제공

- Sprite Took

- Sprite 생성기 : 2D Scene에서 Placeholder가 필요할 경우 사용
- Sprite 에디터 : 큰 이미지에서 Sprite 그래픽스를 추출, 이미지 에디터에서 하나의 텍스처 내에 여러 개의 컴포넌트 이미지를 편집 (예 : 캐릭터의 팔, 다리, 몸을 하나의 이미지 내에 각각 분리된 요소로 유지)
- Sprite Renderer: 2D Sprite를 rendering
- Sprite Packer : 비디오 메모리 사용 및 성능 최적화

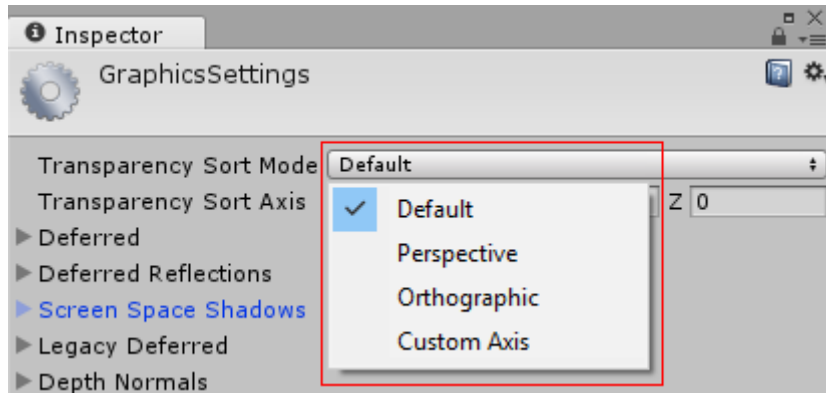
- Importing sprite and set up

- Sprite는 Asset의 한 형태임. 따라서 Asset import하는 절차와 동일 (파일 브라우저 or Assets/import New Asset)
- 프로젝트가 2D 모드로 설정되어 있으면, 이미지는 자동으로 Sprite로 설정
- 프로젝트가 3D 모드로 설정되어 있다면, 해당 Asset의 texture type을 변경해야 함



02. Sprite / Sorting Sprite

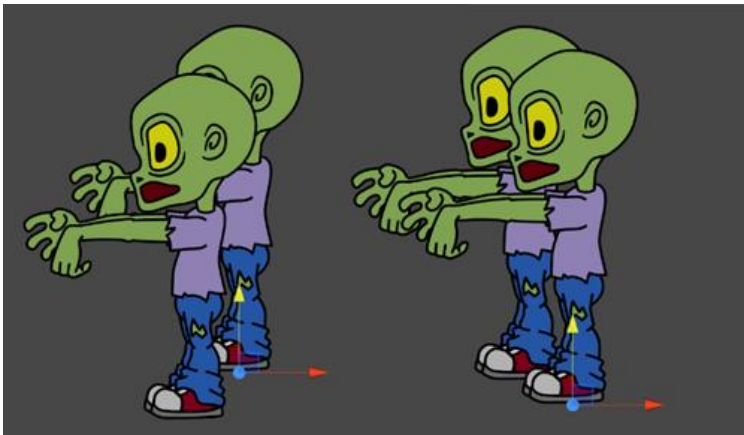
- Sorting sprites
 - Unity Renderer는 레이어 순서, 카메라와의 거리 등에 따라 정렬됨
 - Unity 그래픽스 설정 기능(Edit > Project Settings > Graphics) 에서 Sprite와 카메라의 관계에 따라 프로그래머가 정렬을 제어할 수 있도록 하는 “Transparency Sort Mode” 메뉴를 제공함



카메라의 프로젝션 모드에 따라 sort
Perspective view를 기준으로 sort
Orthographic view를 기준으로 sort
특정 축을 기준으로 sort

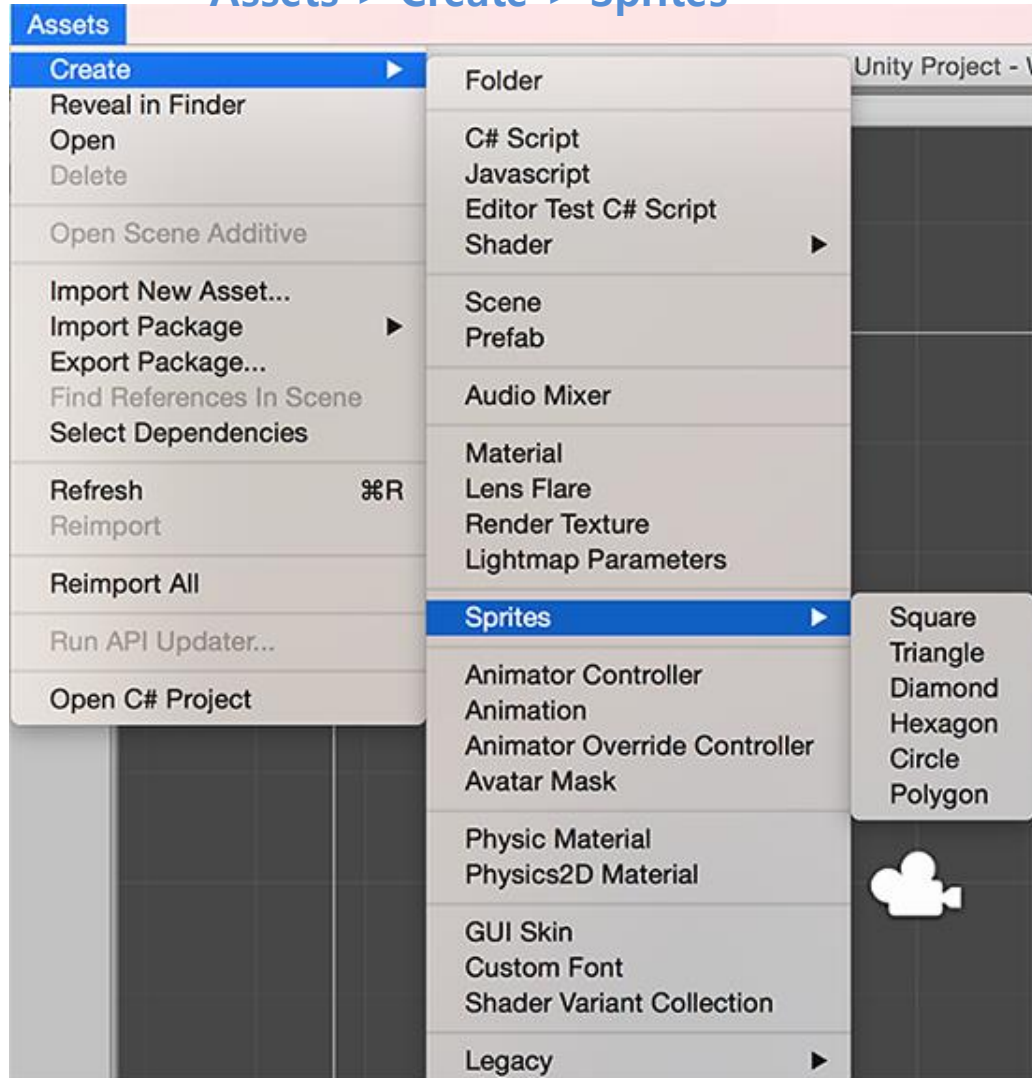
Transparency Sort Axis X Y Z
0 1 0

Y축을 기준으로 정렬

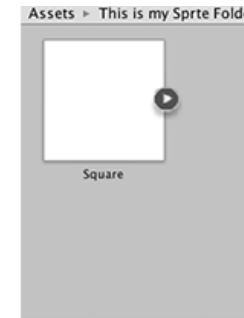
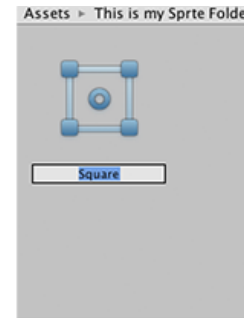


02. Sprite / Sprite Creator

Assets > Create > Sprites

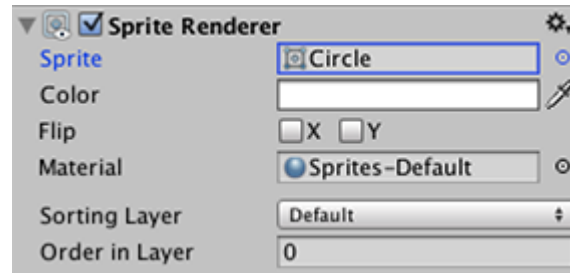


- Sprite Creator
 - 임시 Placeholder sprite(2D) 그래픽스를 생성 (향후 그래픽으로 교체)
 - 종류 : 사각형, 삼각형, 마름모, 육각형, 다각형
 - 결과가 바로 Scene에 적용되는 것이 아니라 Asset 폴더에 생성됨
- Sprite 사용
 - 새 Placeholder sprite는 현재 열린 asset 폴더에 흰색으로 나타남
 - 이름은 변경 가능
 - Scene에 적용하려면 Scene View 또는 Hierarchy로 드래그



02. Sprite / Sprite Creator

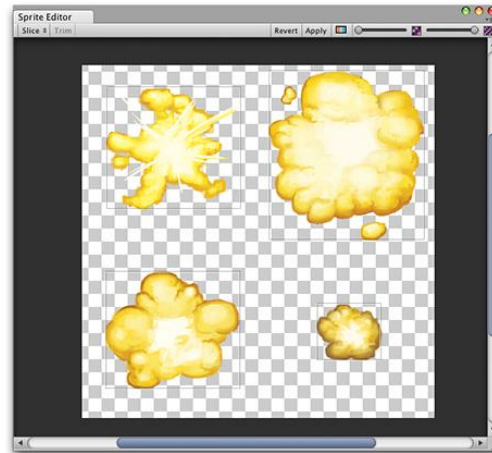
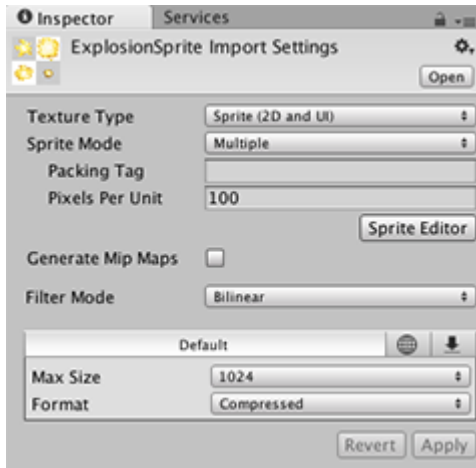
- Placeholder sprite 교체
 - Scene View에서 해당 sprite를 선택 -> Inspector에 있는 Sprite Renderer 를 수정
 - (방법 1) Drag and drop (방법2) 입력 필드의 오른쪽에 있는 작은 동그라미를 클릭한 후 적절한 요소를 선택



- 세부 정보
 - Sprite creator는 4X4 흰색 PNG 아웃라인 텍스처를 생성함
 - Placeholder sprite는 알고리즘에 따라 생성되는 기본 다각형

02. Sprite / Sprite Editor

- Sprite texture에 관련된 여러 그래픽스를 하나의 이미지로 합치는 것이 유리
예) 바퀴만 움직이는 자동차
- 이를 위해 unity에서는 sprite editor를 제공하며, 이 때 편집하고자 하는 이미지의 texture type은 반드시 2D이어야 하며, Sprite mode를 multiple로 설정해야 함
- Sprite Editor 열기
 - 편집하고자 하는 2D 이미지를 Project View에서 선택 (Scene View가 아님을 주목)
 - Texture Import Inspector에 있는 Sprite Editor를 클릭하면 Sprite Editor가 나타남



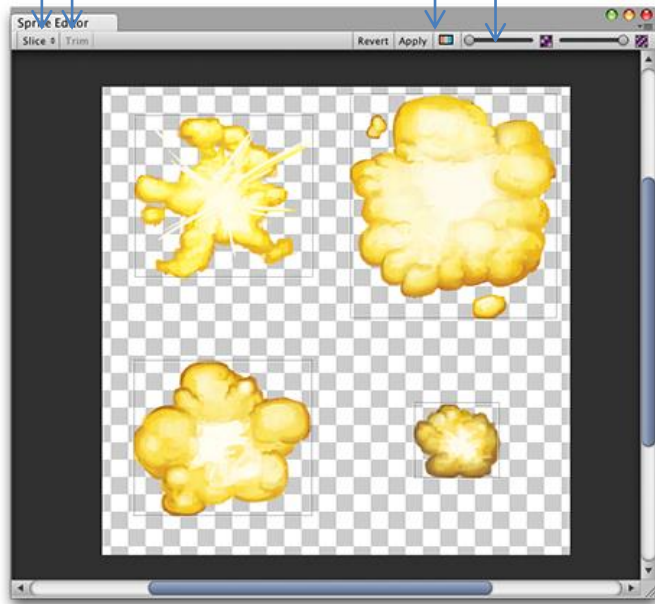
02. Sprite / Sprite Editor

이미지의 요소를 자동분리
(이를 자동 슬라이싱이라 함)

이미지 혹은 알파 레벨

수동영역 선택 시
남는 부분 잘라내기

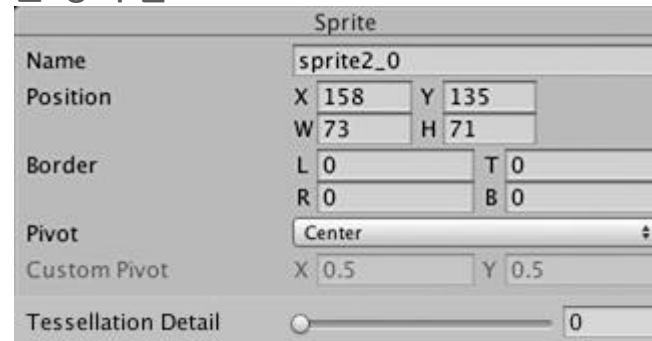
줌



- Sprite editor 수동 구분 방법

- 이미지를 클릭하면 모서리에 핸들이 있는 사각형 영역이 나타남
- 핸들 또는 사각형의 변을 드래그하여 크기 조절
- 하나의 요소를 사각형 영역으로 구분하면 그 다음 영역을 생성할 수 있음 (이름 끝에 영역 식별자로 _숫자가 들어감)

- 영역이 생성되거나, 이미 생성된 영역을 선택하면, 해당 영역을 설정할 수 있도록 작은 창이 뜬다

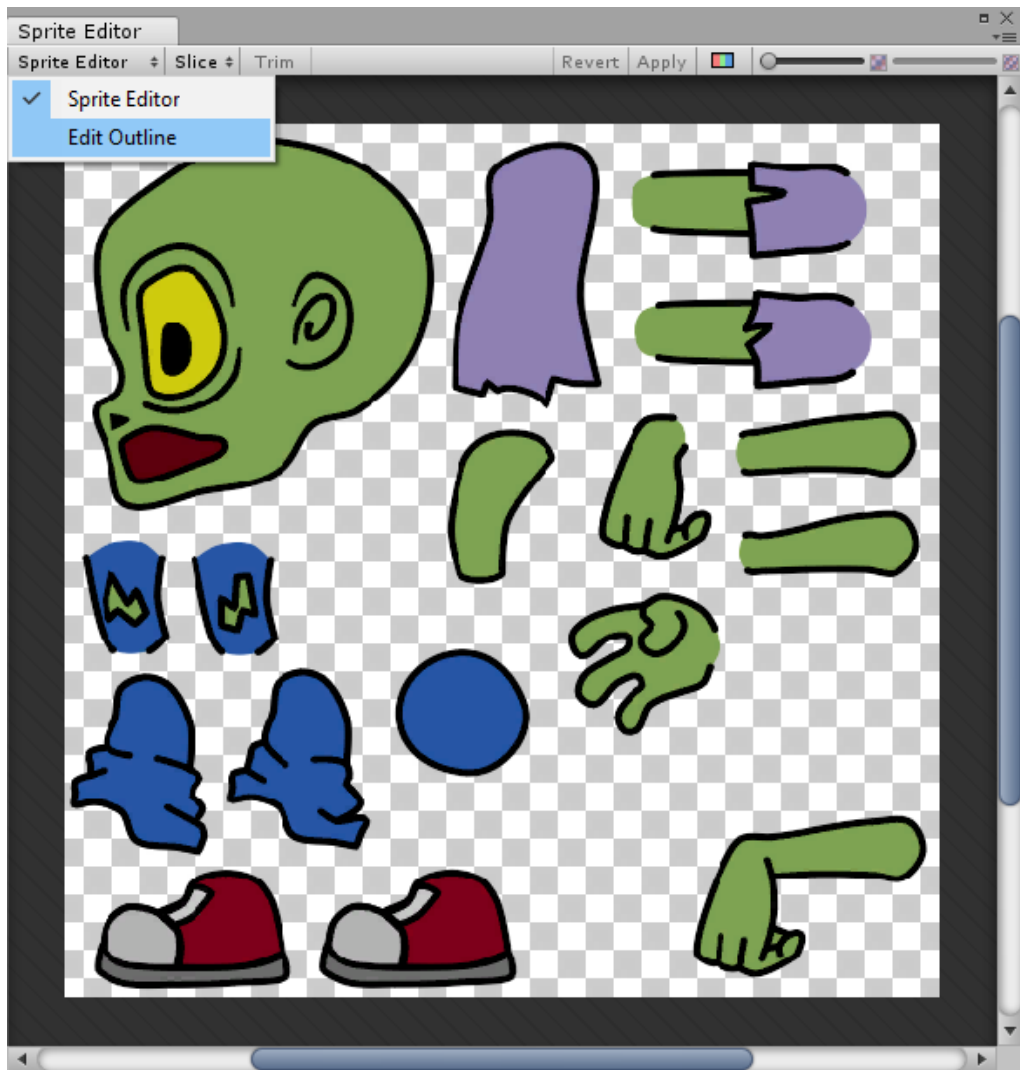


- Auto Slicing

- Slicing type을 automatic으로 설정하면 투명도를 기준으로 각 영역을 자동으로 구분
- 기존 sprite 사각형 영역을 처리하는 방법에 따라 Method 정의 (deleting existing, smart, safe)
Smart : 기존 유지하거나 줄이면서 새로 자르기, Safe : 기존은 보존하면서 자르기

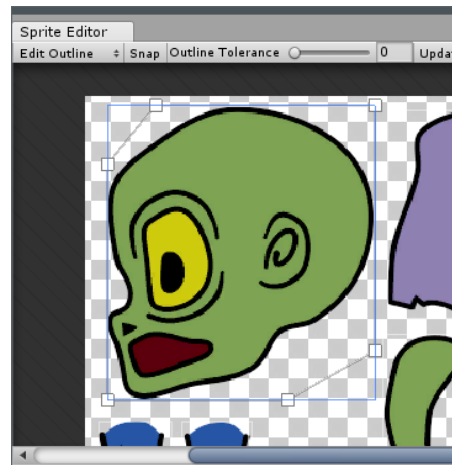
- 폴리곤에서도 사용

02. Sprite / Sprite Editor



- Edit outline

- Sprite이 생성된 공간의 투명한 부분을 줄이는 것이 유리함
- Edit outline 기능은 이러한 투명 공간을 최소화하기 위해 사용함
- 특정 영역을 선택 한 후, Edit outline 메뉴 선택
- 선택된 영역의 선에 마우스를 두면 파란색 사각형이 생성되어 투명 영역을 최소화 할 수 있도록 함



02. Sprite / Sprite Packer

- Sprite를 디자인할 때 각 캐릭터마다 별도의 텍스처 파일로 작업하는 것이 편리함
 - But, 각 그래픽 요소 사이의 빈 공간이 texture의 상당부분을 차지하여 런타임 시의 비디오 메모리 낭비
 - Unity에서는 여러 sprite를 atlas라는 단일 texture에 최대한 모아 넣는 방법을 제공
- Unity의 Sprite Packer는 개별 sprite texture로 atlas를 자동 생성하는 유틸리티
 - Unity가 내부적으로 atlas의 생성 및 사용을 처리하므로 사용자가 수동 할당할 필요가 없음
 - 플레이 모드에 들어갈 때, 혹은 빌드 중에 atlas를 패킹하는 옵션이 있음
 - sprite object용 그래픽스는 일단 atlas가 생성되면 atlas에서 얻어옴
 - Texture의 sprite를 패킹하도록 하려면 texture importer의 packing tag를 지정해야 함

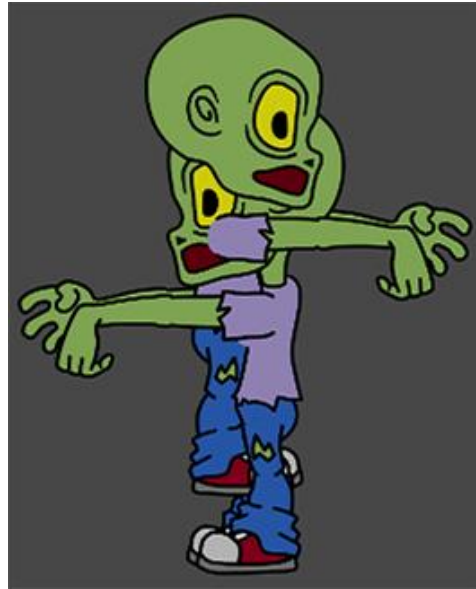


02. Sprite / Sprite Packer

- 패킹 정책
 - Sprite packer는 packing policy에 따라 sprite를 atlas에 할당하는 방법을 결정
 - 옵션 : Default Packer Policy, Tight Packer Policy, Tight Rotate Enabled Sprite Packer Policy
 - Default Packer Policy : packing tag에 [TIGHT]로 설정되어 있지 않으면 디폴트로 사각 패킹을 사용
 - Tight Packer Policy : sprite가 타이트 mesh를 가지고 있을 경우 디폴트로 tight / 만약 [RECT]가 지정되면 사각 패킹
 - Tight Rotate Enabled Sprite Packer Policy : sprite가 tight mesh를 가지고 있으면 디폴트로 tight, 그리고 회전 가능 / 만약 [RECT]가 지정되어 있으면 사각 패킹이 이루어짐
 - Texture importer의 packing tag 속성은 sprite가 packing될 때 atlas의 이름을 직접 선택 (같은 tag를 가지면 같이 packing 된다는 의미)
- Sprite packer customizing
 - 대부분 default packer policy 사용 /
 - 그럼에도 불구하고 custom policy를 구현해야겠다면, UnityEditor.Sprites.IPackerPolicy 인터페이스를 직접 구현필요.
 - 구현 필요한 method
 - GetVersion : Packer의 버전 값을 return (버전이 수정되었음을 알려야 함)
 - OnGroupAtlases : Packing logic을 구현
 - 세부 사항은 첨부된 소스 코드 참조

02. Sprite / Sorting Group

- Sorting Group
 - Sprite renderer의 rendering 순서를 변경하는 컴포넌트
 - 레이어의 순서나 카메라와의 거리를 기준으로 정렬



같은 좀비를 여러 개 붙인 경우

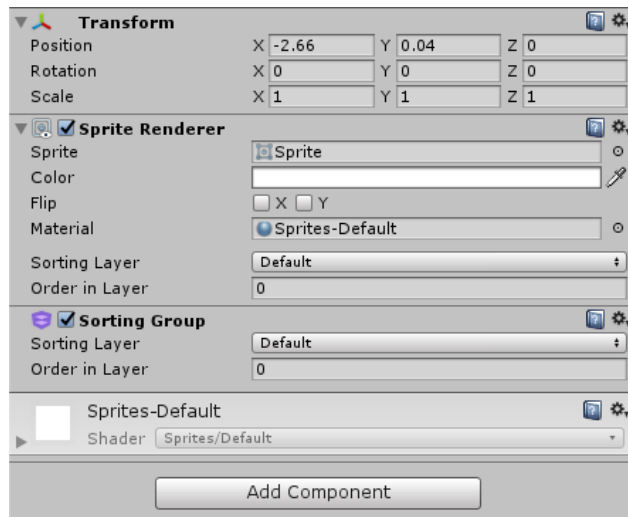
Overlay priority 등에 대한 설정이 없어
일관성 없이 rendering되는 상황 발생



Sorting Group 컴포넌트를 추가한 경우

02. Sprite / Sorting Group

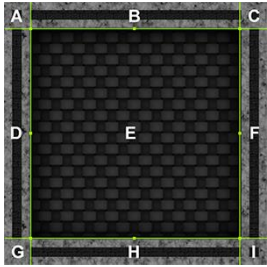
- Sorting Group의 설정
 - GameObject의 root에 Sorting Group 컴포넌트를 추가해야 함
GameObject의 root를 선택하고 Component > Rendering > Sorting Group 선택
 - Sorting Group은 Scene View에서는 볼 수 없음(시각적 표시가 없음)
 - Root GameObject와 그 자녀들은 함께 rendering 됨
- Sorting Group에 대한 sorting
 - 먼저 sorting layer를 기준으로 정렬 / 같은 layer내에서는 overlay priority로 정렬
 - 낮은 순위의 sorting layer는 높은 순위에 의해 가려짐 (낮은 순위가 먼저 rendering된다는 의미)



Rendering 시 해당 sprite의 overlay priority를 정의하는데 사용되는 레이어를 선택
레이어에서 이 sprite의 overlay priority를 설정 (낮은 숫자가 먼저 rendering)

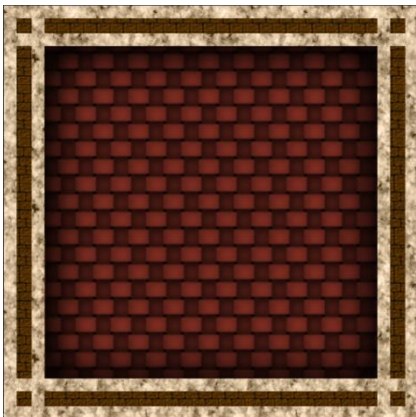
02. Sprite / 9-Slicing Sprites

- 9 slicing : 다양한 크기의 이미지가 필요한 경우, 크기별로 이미지를 만들 필요 없이 하나의 이미지를 늘려서 재사용할 수 있도록 제공하는 2D 기술

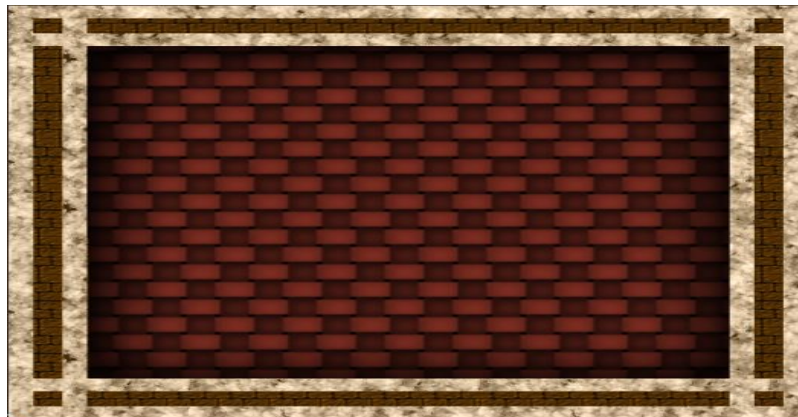


- 네 모서리 (A, C, G, I) 부분은 크기가 변경되지 않음
- 가로변(B, H)은 가로로 길어질 수 있음 / 타일이라면 더 붙임
- 세로변(D, F)은 세로로 길어질 수 있음 / 타일이라면 더 붙임
- 내부영역(E)은 가로 및 세로로 늘어날 수 있음 / 타일이라면 더 붙임

<원본 이미지>



<가로로 늘린 일반 이미지>



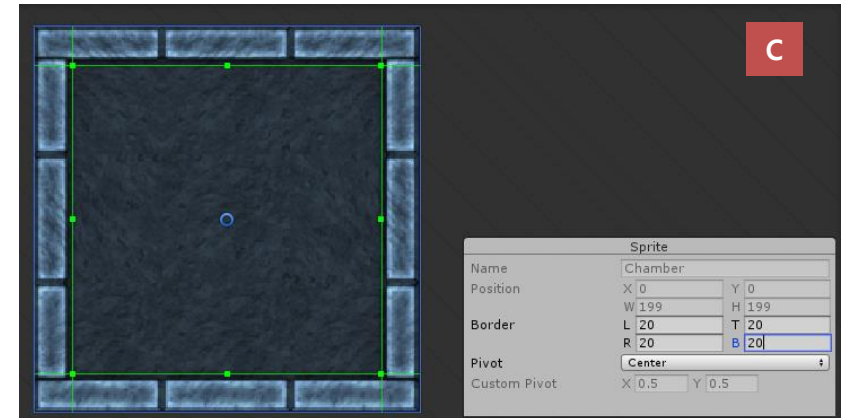
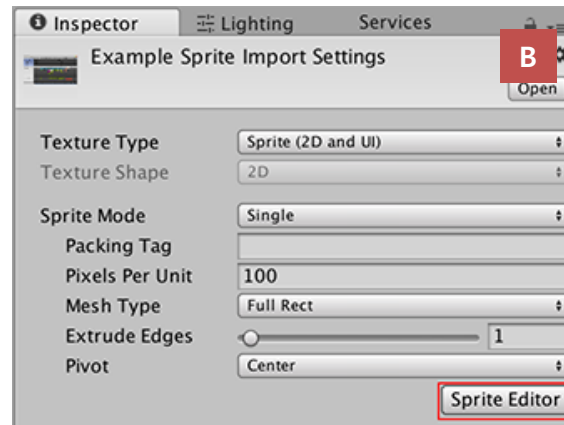
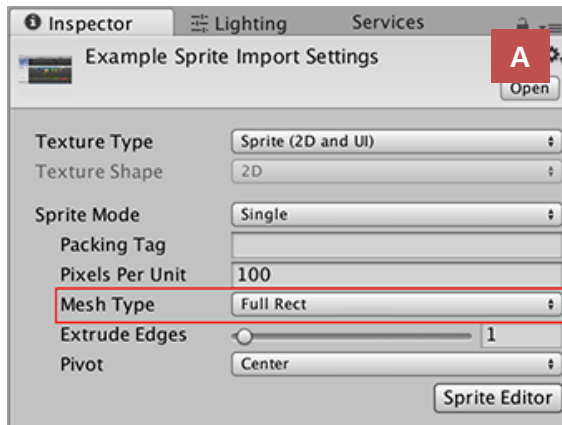
<가로로 늘린 9 slicing image>



02. Sprite / 9-Slicing Sprites

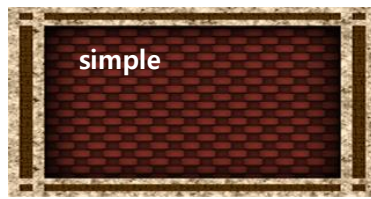
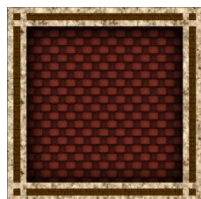
- Setting up 9 Slicing a Sprite

- A. Mesh type이 Full Rect로 설정되어 있는지 확인 / Project e window > Asset에서 해당 sprite을 선택
- B. Sprite Editor 창 실행
- C. sprite의 테두리 정의



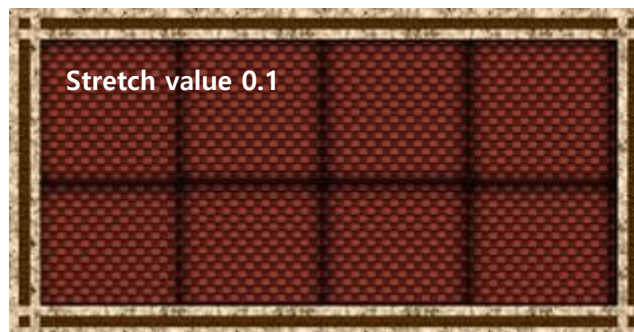
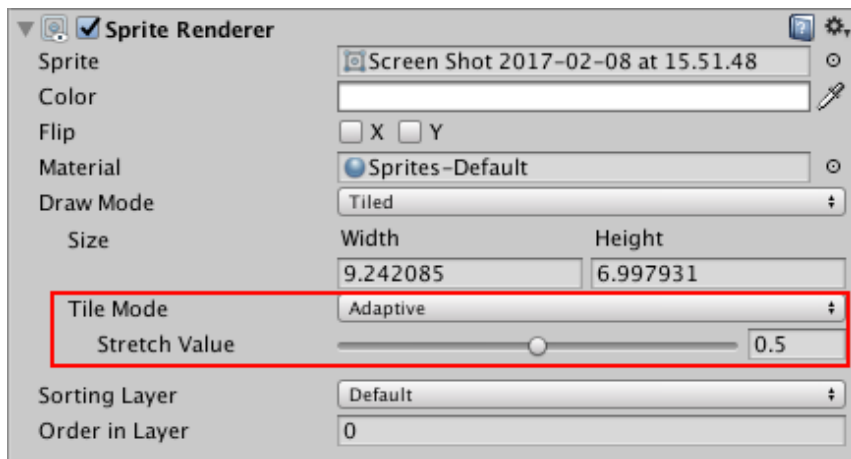
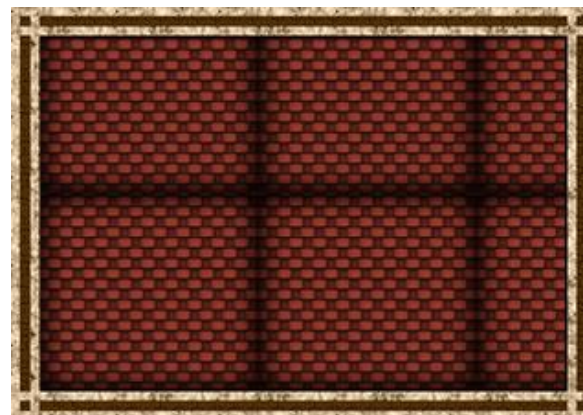
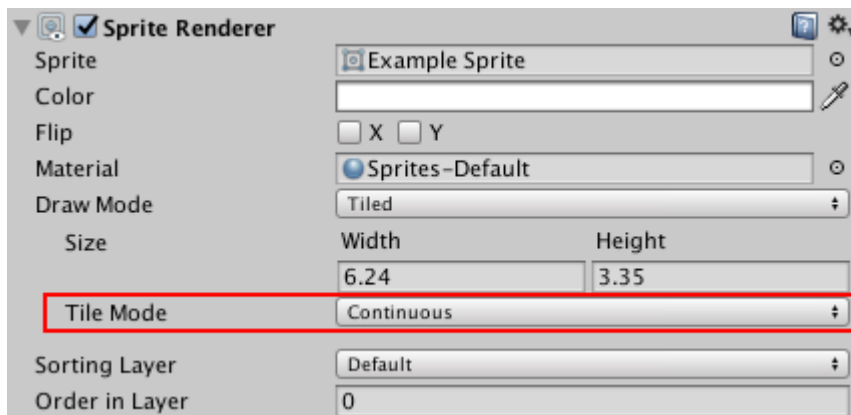
- 9 slicing the Sprite

- Scene View 혹은 Hierarchy window에서 sprite 선택
- Inspector에서 Sprite Renderer의 Draw Mode 변경 → Sliced or Tiled
- Inspector에서 scale을 한번에 변경하면 9 slicing이 적용되지 않음



02. Sprite / 9-Slicing Sprites

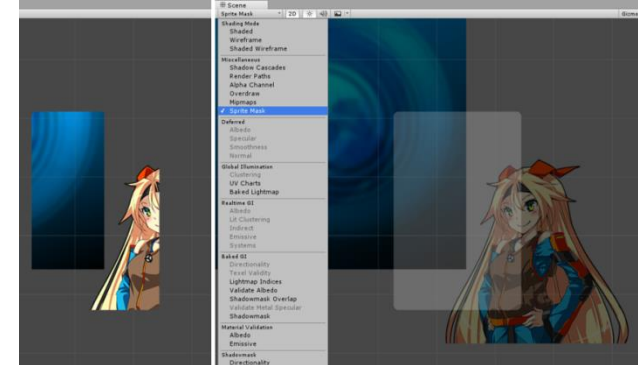
- Tile mode of 9 Slicing



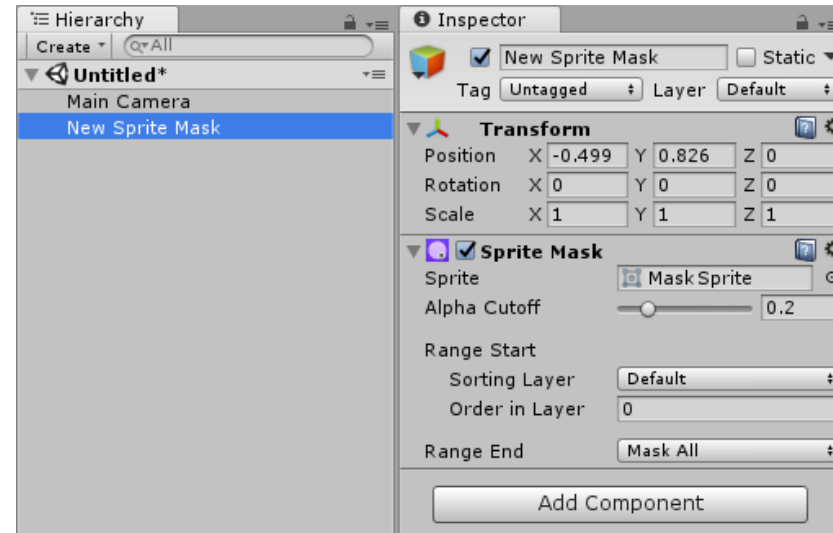
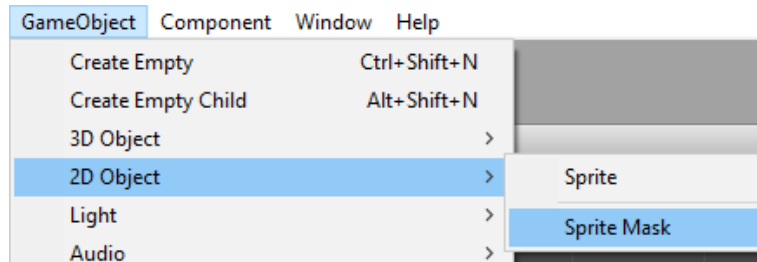
Stretch value가 0.5인 경우 타일은 이미지의 크기가 1배 더 커질 때 마다 반복

02. Sprite / Sprite Mask

- Sprite Mask // 2017.1의 새로운 기능
 - Sprite의 일부를 숨기거나 보여주하고자 할 때 사용 (Sprite Group 포함)
 - Sprite Renderer 컴포넌트를 사용하는 오브젝트에만 영향을 미침

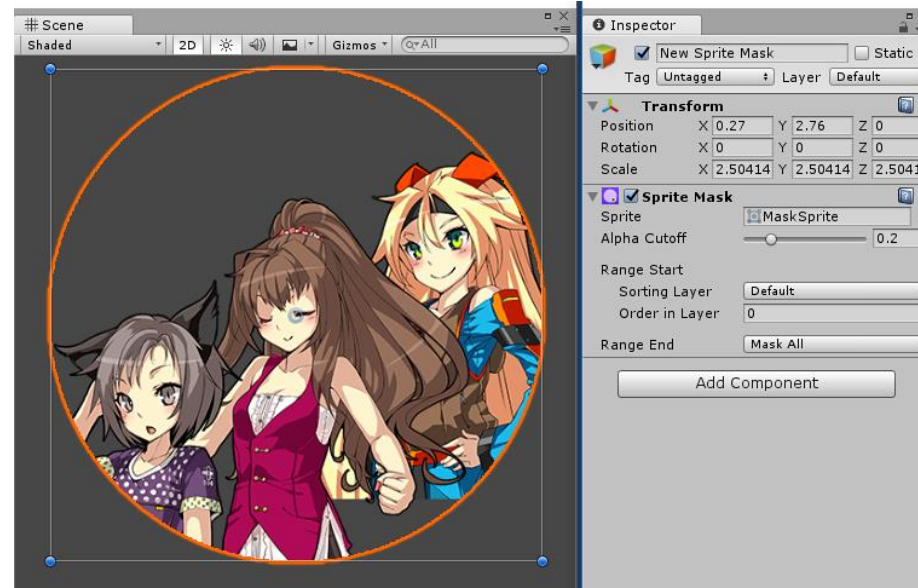
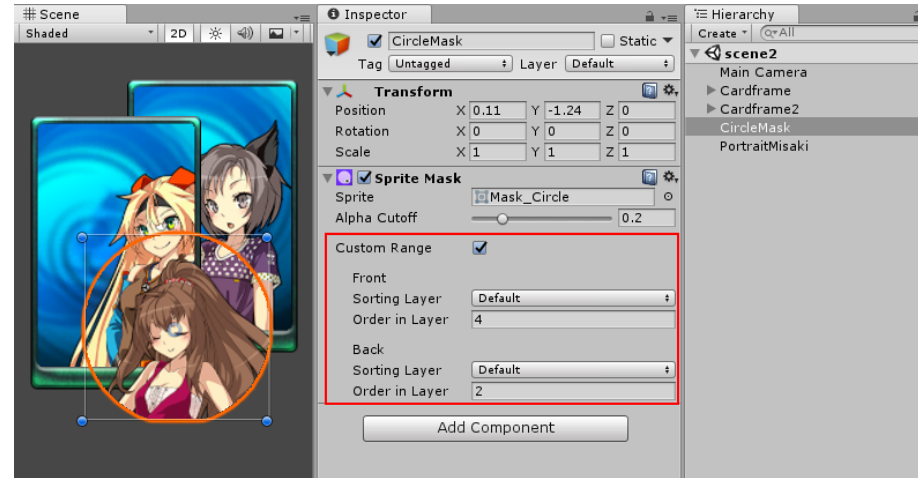
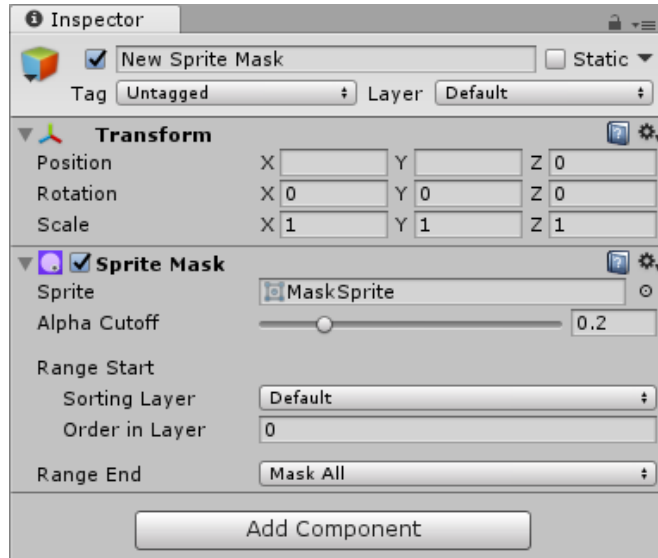


- Sprite Mask의 생성
 - GameObject > 2D Object > Sprite Mask



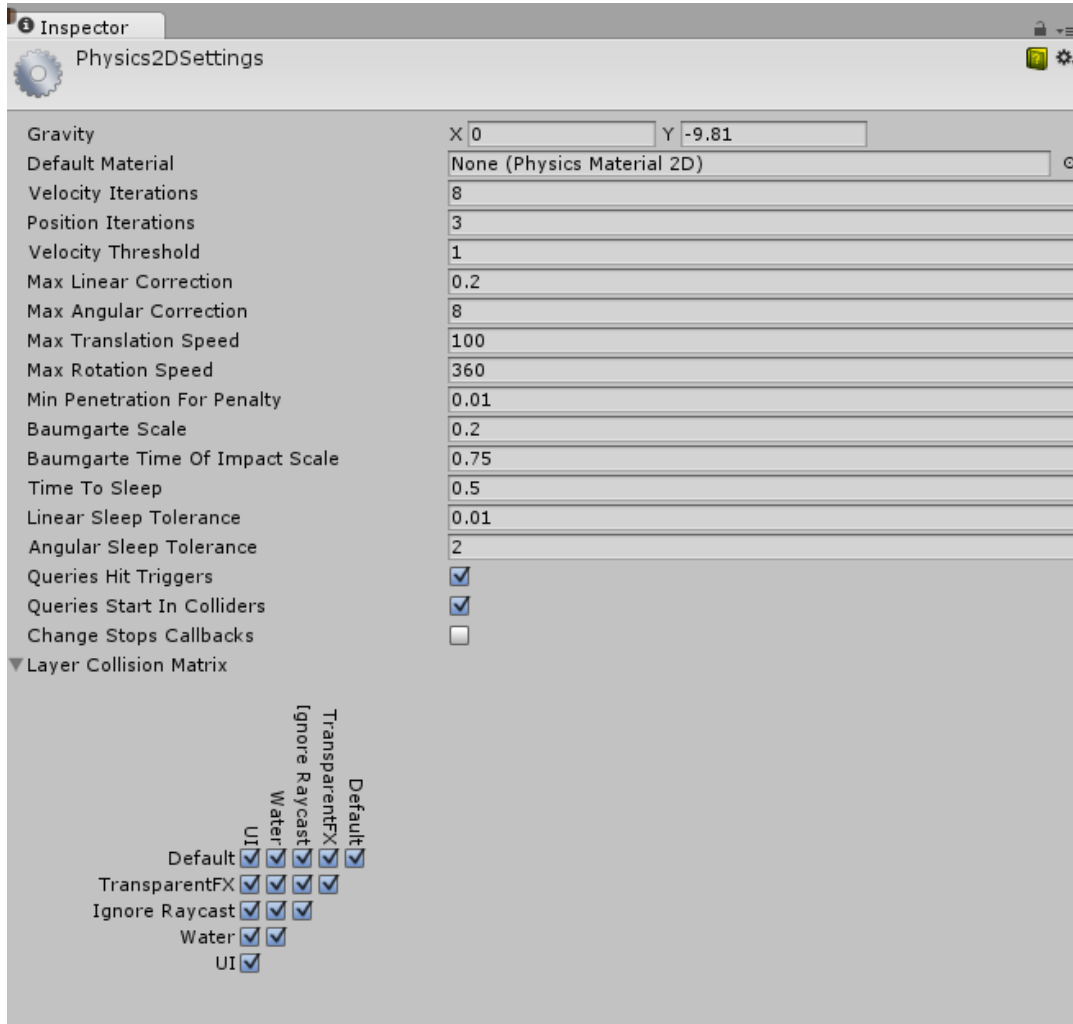
02. Sprite / Sprite Mask

- Sprite Mask



03. Physics 2D

- Physics 2D 전역 설정 (Edit > Project Settings > Physics 2D)



Gravity 중력 : Y축 - 방향으로 9.81

Gravity는 Rigidbody 2D(강체)에 모두 적용. Rigidbody는 Component로 추가해야 함

Default Material : Collider 2D가 정의되지 않은 경우 적용할 default - None

Velocity Iterations : 속도 효과(velocity effect)에 대한 물리 엔진 반복연산횟수

숫자가 높으면 정확도가 높아지지만 CPU 시간도 함께 늘어남

Position Iterations : 위치 변화(position changes)에 대한 물리 엔진 반복연산횟수

숫자가 높으면 정확도가 높아지지만 CPU 시간도 함께 늘어남

Max Linear Correction : linear position correction 최대값

Max Angular Correction : linear angular correction 최대값

Max Translation Speed : 일반 2D GameObject의 최대 linear speed

Max Rotation Speed : 일반 2D GameObject의 최대 rotation speed

Min Penetration For Penalty : 최소 접촉 침투 반경

Baumgarte Scale : 충돌이 얼마나 빨리 해소되는지 결정

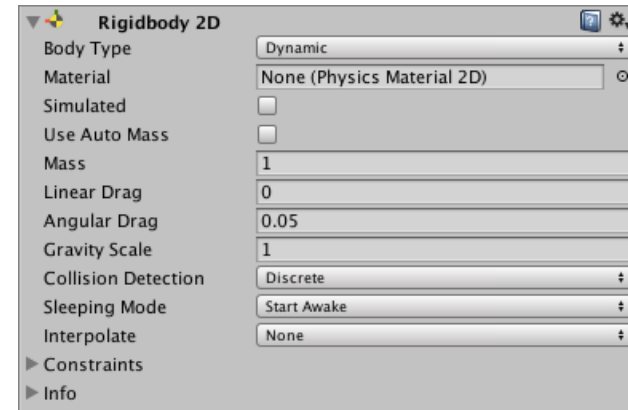
:

필요 시 참조

03. Physics 2D / Rigidbody 2D

- Rigidbody 2D Component

- Object를 물리 엔진이 제어하도록 만듦
- Rigidbody (3d용) component의 많은 개념이 2D에 적용

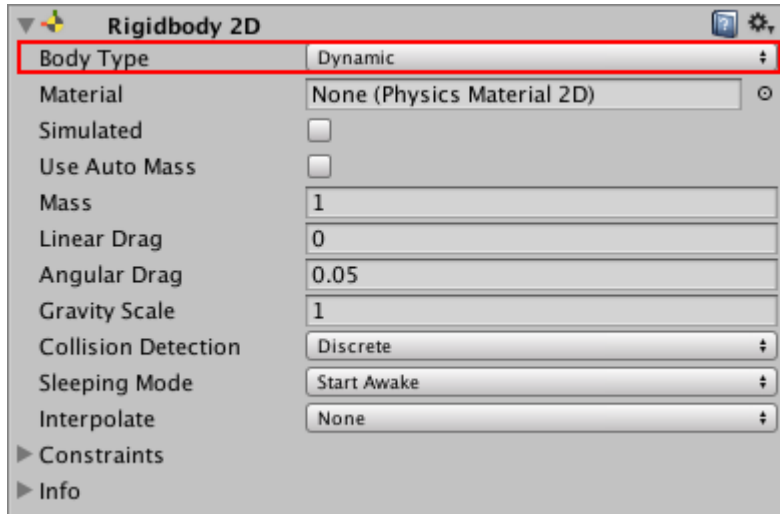


- 작동 방식

- Transform component : GameObject가 Scene 내에서의 배치, 회전, 스케일(크기 변화)되는 방식을 정의
- GameObject간의 움직임에 따른 위치 변화/충돌 등은 결국 Transform의 변경을 가져옴
→ GameObject는 충돌 상황에서의 동작을 구현하기 위해 Collider2D component 를 가짐
- Rigidbody 2D Component는 Physics 2D에 기반한 이러한 변화를 Transform으로 전달하는 방법
→ Rigidbody2D가 Transform(위치, 각도, 스케일)을 override 한다
→ GameObject에 추가된 Collider2D Component는 해당 Object의 Rigidbody2D Component와 묵시적으로 연결

03. Physics 2D / Rigidbody 2D

Rigidbody 설정 / Body Type



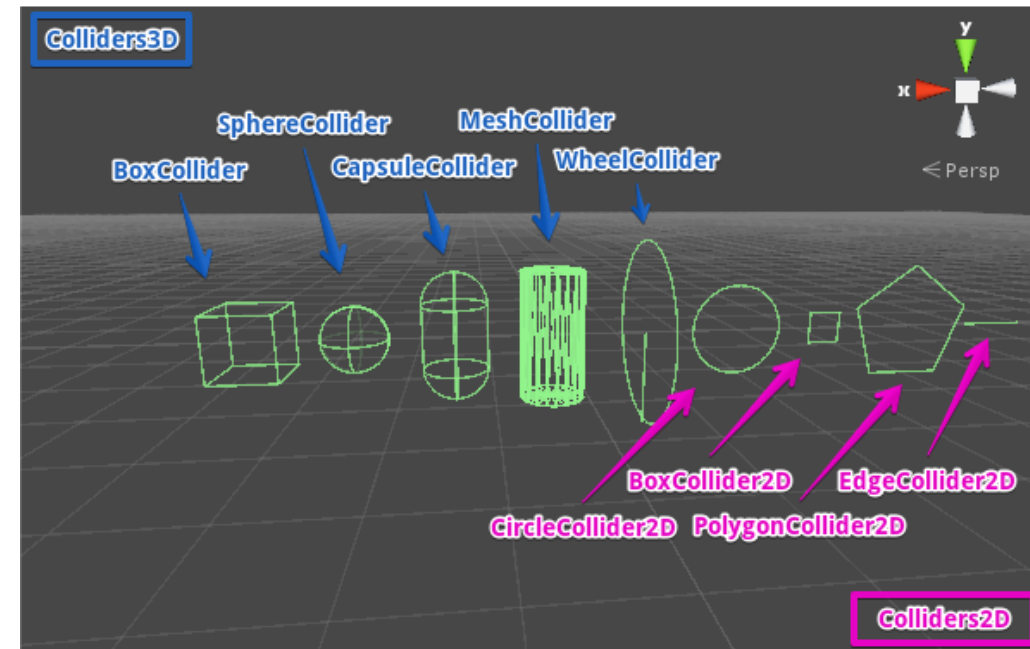
- **Dynamic**
 - 시뮬레이션 상태에서 움직이도록 디자인
 - 중력과 힘에 영향을 받음
 - 다른 body type들과 충돌하며, 가장 interactive
 - Rigidbody의 default body type
- **Kinematic**
 - 명시적인 사용자 제어가 있는 경우를 고려하여 디자인
 - 중력과 힘에 영향을 받지 않음(시스템 리소스를 적게 사용)
 - 런타임에 위치를 이동시키고자 할 경우 스크립트 사용
Rigidbody2D.MovePosition 또는 Rigidbody2D.MoveRotation
 - Dynamic Rigidbody와만 충돌 (default)
모든 2D Rigidbody와 충돌하게 하려면 Use Full Kinematic Contacts 설정
 - 질량이 무한으로 설정되며, 충돌 시 움직일 수 없는 오브젝트 처럼 동작
- **Static**
 - 전혀 움직이지 않도록 디자인 / 리소스를 가장 적게 사용
 - 충돌 시 (질량이 무한하여) 움직일 수 없는 오브젝트처럼 동작
 - Collider 2D가 포함되고 Rigidbody2D가 포함되지 않은 경우에도 static으로 간주
 - 런타임에 Static object를 이동시켜야 할 경우에는 명시적으로 static rigidbody를 사용하는 것이 performance에 유리함 (Unity 디자인 이슈)

03. Physics 2D / Rigidbody 2D

- **Simulated** : 시뮬레이션 사용
Rigidbody 2D 및 연결된 모든 Collider 2D / Joint 2D가 실행 시 상호 작용하도록 하기 위해서는 반드시 체크
- **Use Auto Mass** : Collider 2D가 질량 자동 감지
- **Mass** : 질량을 직접 정의
- **Linear Drag** : 위치 이동에 영향을 미치는 계수
- **Angular Drag** : 회전에 영향을 미치는 계수
- **Gravity Scale** : 중력에 영향을 받는 정도
- **Collision Detection** : 충돌 감지
 - Discrete : 주기적으로 한번씩 감지 (감지 안될 수도 있음)
 - Continuous : 계속 감시 (Unity에서 첫 번째 충돌 지점을 산출하여 그리로 이동, CPU 많이 먹음)
- **Sleeping Mode** : GameObject의 휴식상태 처리방식
 - Never Sleep : 안 잠
 - Start Awake : 깬 상태로 시작
 - Start Asleep : 자는 상태로 시작
충돌이 발생하면 일어남
(Object가 잔다는 것은 그 Object에 대해 CPU를 할당하지 않는다는 의미)
- **Interpolate** : physics update 시, 움직임이 보간되는 방법 정의 (해보고 선택)
 - None : 보간 안 함
 - Interpolate : 이전 프레임에서의 위치를 근거로 스무딩
 - Extrapolate : 다음 프레임에서의 위치를 추정하여 스무딩
- **Constraints** : 움직임에 대한 제약 사항 정의
 - Freeze Position : X, Y 중 Rigidbody를 적용하지 않을 축을 설정 (해당 축을 선택)
 - Freeze Rotation : 회전을 적용하지 않을 경우 Z 축 선택

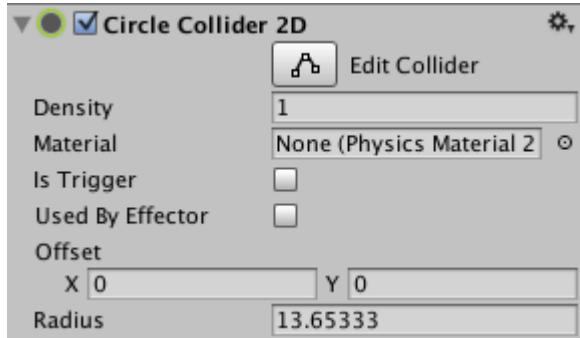
03. Physics 2D / Collider 2D

- Collider 2D : 물리적 충돌 시의 게임 오브젝트의 모양을 정의
- 종류
 - 원형 충돌 영역을 위한 Circle Collider 2D.
 - 정사각형 및 직사각형 충돌 영역을 위한 Box Collider 2D.
 - 자유형 충돌 영역을 위한 Polygon Collider 2D.
 - 자유형 충돌 영역 및 완전히 닫히지 않은 영역(원형 컨벡스 코너 등)을 위한 Edge Collider 2D.
 - 원형 또는 마름모 모양의 충돌 영역을 위한 Capsule Collider 2D.
 - Box Collider 2D 및 Polygon Collider 2D를 병합하기 위한 Composite Collider 2D (복합 Collider).



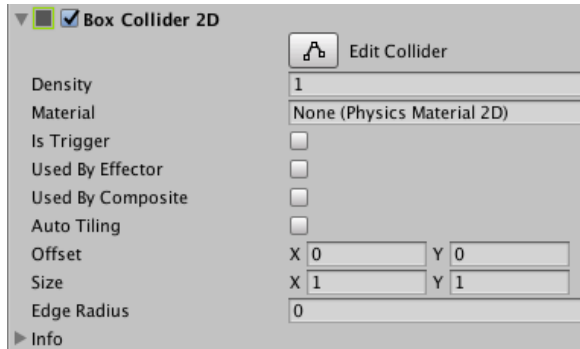
03. Physics 2D / Collider 2D

- Circle Collider 2D



Property	기능:
Density	여기서 밀도를 변경하면 질량에 영향을 줌. 이 값을 0으로 설정할 경우 관련 Rigidbody 2D는 모든 질량 계산 시 써클 Collider 2D를 무시. 이 옵션은 관련 rigidbody 2D의 Use Auto Mass 를 선택했을 경우에만 사용 가능합니다.
Material	마찰이나 바운스와 같은 충돌 Property를 결정하는 물리 머티리얼
Is Trigger	Circle Collider 2D가 트리거처럼 작동하도록 하려면 선택
Used by Effector	Circle Collider 2D가 연결된 이펙터 2D에서 사용되길 원할 경우 이 상자를 선택
Offset	Circle Collider 2D의 offset (X, Y축으로 조금씩 이동 시키고 싶을 때)
Radius	원의 반지름 (local space unit)

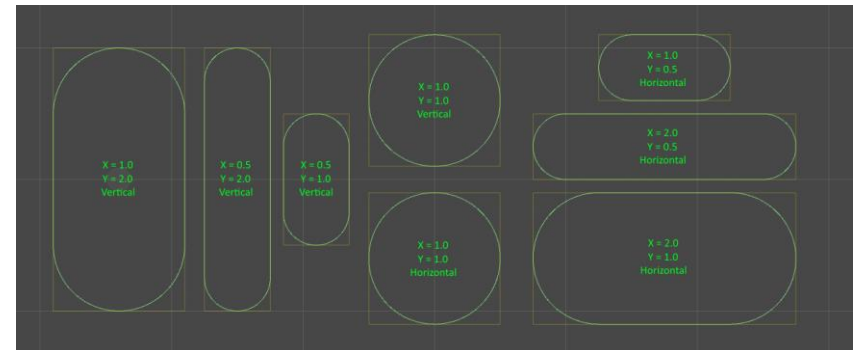
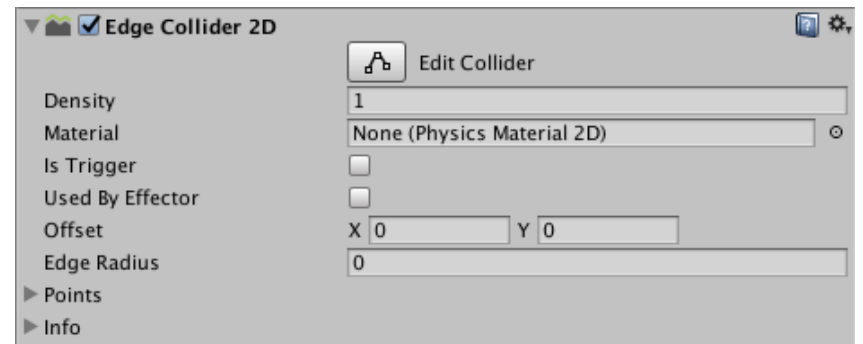
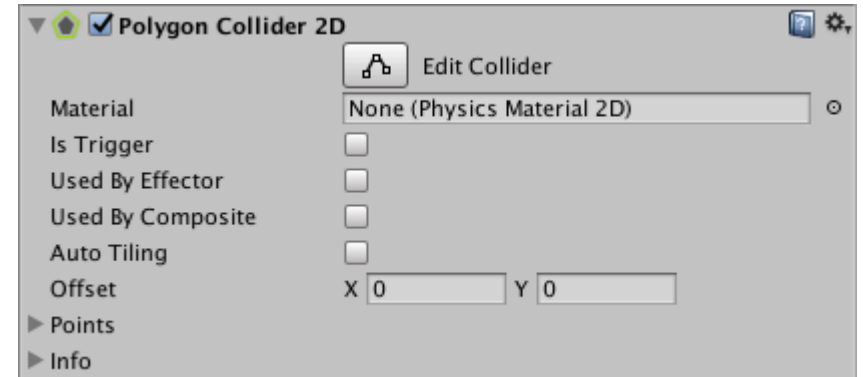
- Box Collider 2D



Property	기능
Density, Material, Is Trigger, Used by Effector, Offset은 위와 같음	
Used by Composite	Collider가 연결된 Composite Collider 2D 에서 사용되도록 하려면 이 체크박스를 선택 Used by Composite 를 활성화하면, 다른 Property는 연결된 복합 Collider 2D에서 제어되기 때문에 박스 Collider 2D 컴포넌트에서 사라짐. (사라지는 속성 : Density, Material, Is Trigger, Used By Effector, Edge Radius)
Auto Tiling	선택된 Sprite Sprite Renderer 컴포넌트의 Draw Mode 가 Tiled 로 설정돼 있을 경우 이 체크박스를 선택해야 함.
Size	박스 크기를 설정 (local space unit)
Edge Radius	모서리 주변이 원모양이 되도록 반지름 조정 (값을 넣으면 Collider의 크기가 커짐)

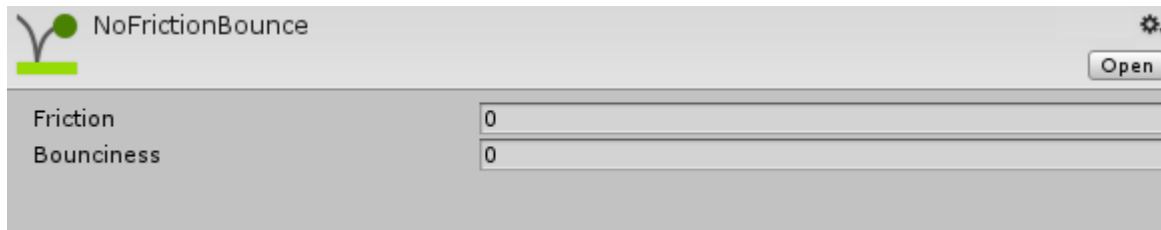
03. Physics 2D / Collider 2D

- Polygon Collider 2D
 - 라인들로 collider를 구성하여 Sprite에 맞게 매우 정밀하게 조정 가능
 - 다른 Property는 Box collider와 대동소이하며,
 - 라인들을 연결할 수 있도록 포인트 입력할 수 있음
 - 각 포인트는 수동으로 입력할 수도 있으며, Unity에서 관련 asset을 분석하여 자동으로 할당하게 할 수도 있음
(해당 asset을 polygon collider 2D component 위에 drag & drop)
- Edge Collider 2D
 - Polygon과 유사하나 영역을 완전히 감쌀 필요가 없음
 - Poly line을 직접 수정하려면 Scene View에서 에지나 꼭지점에 마우스를 올리고 Shift 키를 누르고 있어야 함
- Capsule Collider 2D
 - 수직 또는 수평 방향으로 늘어나게 할 수 있도록 구성



03. Physics 2D / Physics Material 2D

- Physics Material 2D
 - 2D Physics Object간의 충돌이 발생할 때 마찰과 탄성을 조정하기 위해 사용
 - Assets > Create > Physics2D Material



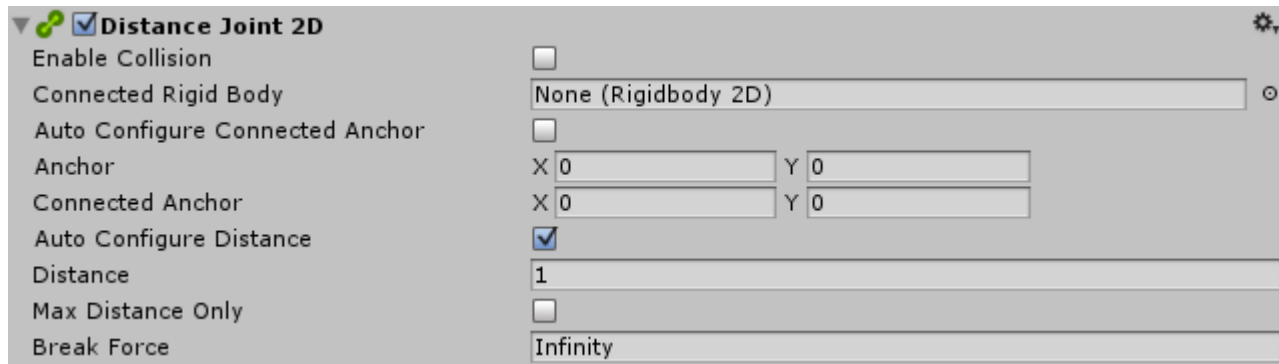
0이면 바운스없고,
1이면 에너지 무손실 바운스

03. Physics 2D / 2D Joints

- 2D Joints
 - GameObject를 하나로 연결하는 기능을 수행
 - 2D에서 사용 가능한 joint는 이름 끝에 2D를 붙임
- 2D Joints의 종류
 - Distance joint 2D / 거리 조인트 : 두 rigidbody object를 일정한 거리로 연결
 - Fixed joint 2D / 고정 조인트 : 두 Object의 상대적 위치를 고정 (위치, 각도 유지)
 - Friction joint 2D / 마찰 조인트 : 두 Object 사이의 속도, 각도 차를 감소
 - Hinge joint 2D / 경첩 조인트 : 회전 중심으로 연결
 - Relative joint 2D / 상대 조인트 : 두 Object가 서로의 위치를 기준으로 유지 (플레이어를 따라 움직이는 펫)
 - Slider joint 2D / 슬라이더 조인트 : 공간에서 선을 따라 미끄러지듯 연결
 - Spring joint 2D / 스프링 조인트 : 스프링이 연결된 것 처럼 두 Object 연결
 - Target joint 2D / 타겟 조인트 : Object가 아니라 특정 target에 연결
 - Wheel joint 2D / 바퀴 조인트 : 바퀴처럼 연결
- 세부 정보
 - Constraints : 각 joint는 제약 사항을 하나 이상 가짐

03. Physics 2D / 2D Joints

- 주요 Property of Distance Joint 2D
 - Enable Collision : 연결된 두 Object 간에 충돌을 활성화
 - Connected Rigid Body : 연결할 Object
만일 None으로 설정하면 Connected Anchor로 지정한 좌표에 연결됨
 - Auto Configure Connected Anchor : 연결 앵커 위치 자동 설정
 - Anchor : Distance Joint 2D와 이 GameObject가 연결될 지점
 - Connected Anchor : Distance Joint 2D와 다른 GameObject가 연결될 지점
 - Auto Configure Distance : 두 GameObject 간 현재 거리를 자동으로 감지하여 이를 계속 유지하도록 설정
 - Distance : 두 Object간 거리 지정
 - Max Distance Only : 최대 거리만 적용
 - Break Force : Distance Joint를 파괴하고 삭제하는데 필요한 힘의 수준을 지정

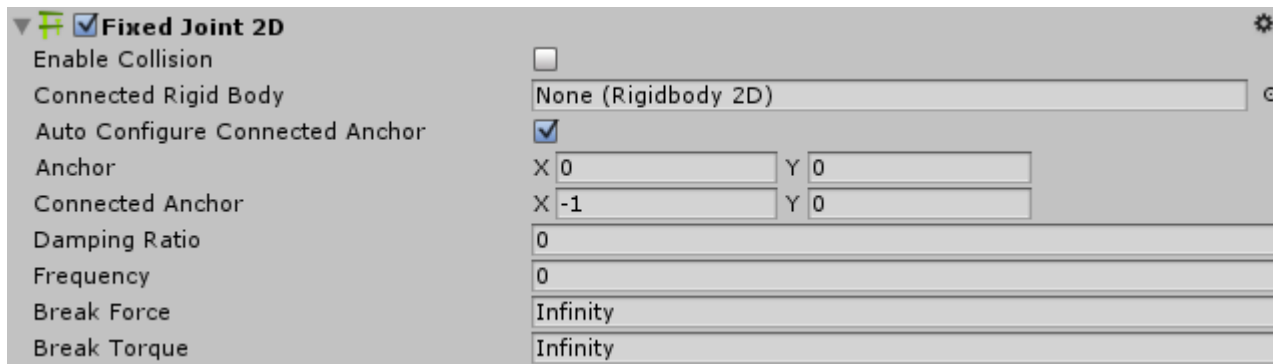


The screenshot shows the Unity Inspector for a **Distance Joint 2D** component. The component is selected, and its properties are visible. The **Enable Collision** checkbox is unchecked. The **Connected Rigid Body** dropdown is set to **None (Rigidbody 2D)**. The **Auto Configure Connected Anchor** checkbox is unchecked. The **Anchor** and **Connected Anchor** fields are both set to **X 0 Y 0**. The **Auto Configure Distance** checkbox is checked. The **Distance** field is set to **1**. The **Max Distance Only** checkbox is unchecked. The **Break Force** field is set to **Infinity**.

Property	Value
Enable Collision	<input type="checkbox"/>
Connected Rigid Body	None (Rigidbody 2D)
Auto Configure Connected Anchor	<input type="checkbox"/>
Anchor	X 0 Y 0
Connected Anchor	X 0 Y 0
Auto Configure Distance	<input checked="" type="checkbox"/>
Distance	1
Max Distance Only	<input type="checkbox"/>
Break Force	Infinity

03. Physics 2D / 2D Joints

- Fixed Joint 2D : 두 Object 간의 연결을 Fix
 - Spring 처럼 Fix 할 수 있음
- Property of Fixed Joint 2D
 - Enable Collision / Connected Rigidbody / Auto Configure Connected Anchor / Anchor / Connected Anchor는 Distance Joint 와 동일
 - Damping Ratio : 스프링 진동을 억제하는 정도, 값이 클수록 적게 움직임
 - Frequency : 초당 스프링 진동수
1~1,000,000 값으로 값이 클수록 스프링이 경직됨, 단 0이면 완전히 경직
 - Break Force : Joint를 파괴하는데 필요한 힘의 수준
 - Break Torque : Joint를 파괴하는데 필요한 토크(회전력)의 수준



Relative와의 차이 :

- Relative는 스프링이 아니라 모터를 사용
- Fixed Joint 2D는 Anchor point와 동작, Relative는 Anchor point가 없음
- Relative는 실시간으로 Object간 위치 및 각도를 변경할 수 있음

03. Physics 2D / 2D Joints

- Friction Joint 2D : 마찰로 이동이나 회전을 줄이게 되는 Joint
 - Enable Collision / Connected Rigidbody / Auto Configure Connected Anchor / Anchor / Connected Anchor는 Distance Joint 와 동일
 - Max Force : 연결된 Object의 직선 운동을 설정. 높을 수록 저항이 큼
 - Max Torque : 연결된 Object의 회전 운동을 설정. 높을 수록 저항이 큼
 - Break Force, Break Torque는 Fixed Joint와 동일
- Hinge Joint 2D : 경첩
 - Enable Collision / Connected Rigidbody / Auto Configure Connected Anchor / Anchor / Connected Anchor는 Distance Joint 와 동일
 - Use Motor : Hinge Motor를 활성화
 - Motor Speed : 초당 각도
 - Maximum Motor Force : 최대 모터 토크
 - Use Limits : 회전 각도를 제한
 - Lower Angle : 각도 하한
 - Upper Angle : 각도 상한
 - Break Force, Break Torque는 Fixed Joint와 동일

03. Physics 2D / 2D Joints

- Relative Joint 2D : 서로의 위치에 근거해서 포지션 유지
 - Enable Collision / Connected Rigidbody / Auto Configure Connected Anchor / Anchor / Connected Anchor는 Distance Joint 와 동일
 - Max Force : 연결된 Object의 직선 거리를 설정. 값이 클 수록 Offset 유지를 위해 큰 힘을 사용
 - Max Torque : 연결된 Object의 각도를 설정. 값이 클 수록 Offset 유지를 위해 큰 힘을 사용
 - Correction Scale : Joint 미세 조정
 - Auto Configure Offset : Object 사이의 거리와 각도를 자동으로 설정 및 유지
 - Linear Offset : 연결된 Object 간의 거리를 지정, 유지
 - Angular Offset : 연결된 Object간의 각도를 지정, 유지
 - Break Force, Break Torque는 Fixed Joint와 동일
- Slider Joint 2D : 연결된 GameObject가 공간에서 선을 따라 미끄러지도록
 - Enable Collision / Connected Rigidbody / Auto Configure Connected Anchor / Anchor / Connected Anchor는 Distance Joint 와 동일
 - Auto Configure Angle : 두 Object 간의 각도를 자동으로 설정
 - Angle : Joint의 각도를 지정
 - Use Motor : 슬라이딩 모터를 사용할 경우 체크
 - Motor Speed, Maximum Motor Force은 hinge joint와 동일
 - Use Limits : 직선 이동 힘에 대한 제한이 있어야 한다면 체크
 - Translation / Lower Translation / Upper Translation : Object와 Connected Anchor Point 사이의 거리 제한
 - Break Force, Break Torque는 Fixed Joint와 동일

03. Physics 2D / 2D Joints

- Spring Joint 2D : 두 GameObject가 Spring 처럼 붙어 있도록
 - Property는 Fixed Joint와 동일 / 다만 움직이는 Object에 적용
- Target Joint 2D : 특정 Target에 연결되는 Spring Type Joint
 - Property는 다른 joint 대비 심플
- Wheel Joint 2D : joint에 motor 적용
 - Slider Joint 2D와 hinge joint 2D를 조합한 것 처럼 움직임
 - Spring 관련 property와 motor 관련 properties가 혼합되어 있음

