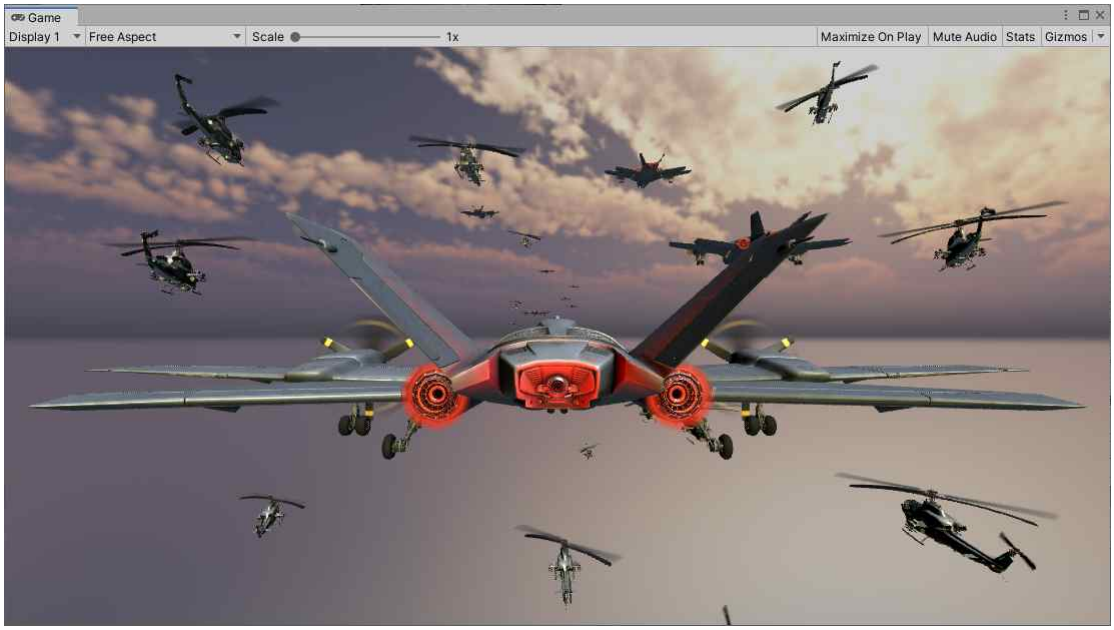
3. 샘플 프로젝트(Sample Project)

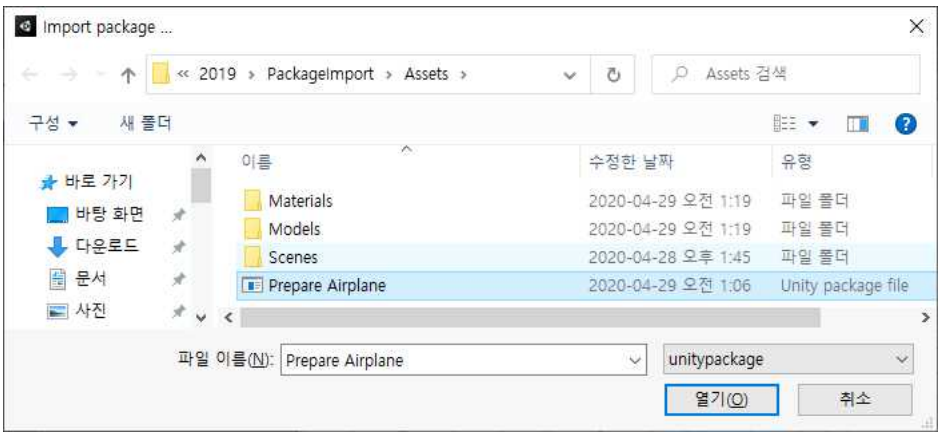
(1) Airplane

① 준비하기

다음과 같은 예제 프로젝트를 구현하자.

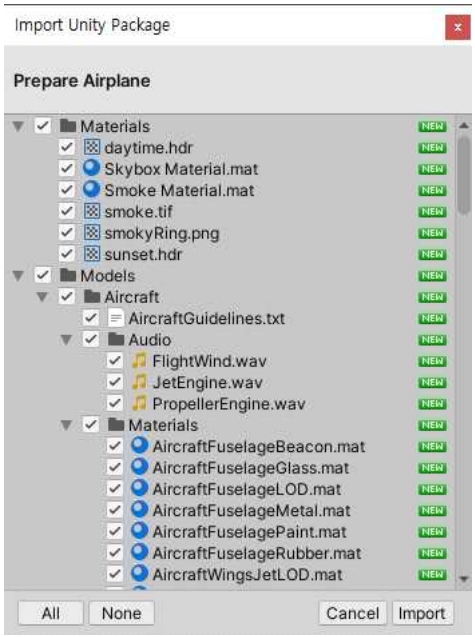


- 프로젝트 생성하기
  - ❶ 새로운 프로젝트를 생성한다.
  - ❷ eClass에서 “Prepare Airplane.unitypackage“ 파일을 다운로드한다.
    유니티 패키지 파일에 프로젝트에 사용할 모델, 재질 등을 미리 담아 놓았다.
  - ❸ 메뉴 “Assets ▶ Import Package ▶ Custom Package...“를 선택한다.

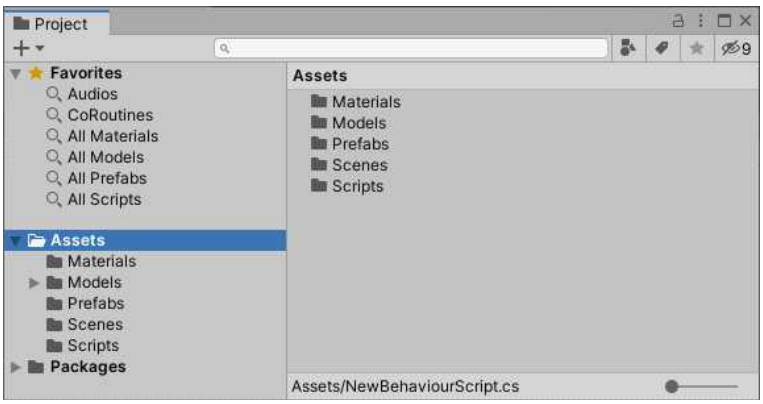


  - ❹ 다운로드한 “Prepare Airplane.unitypackage“ 파일을 선택하고 ”열기“를 선택한다.

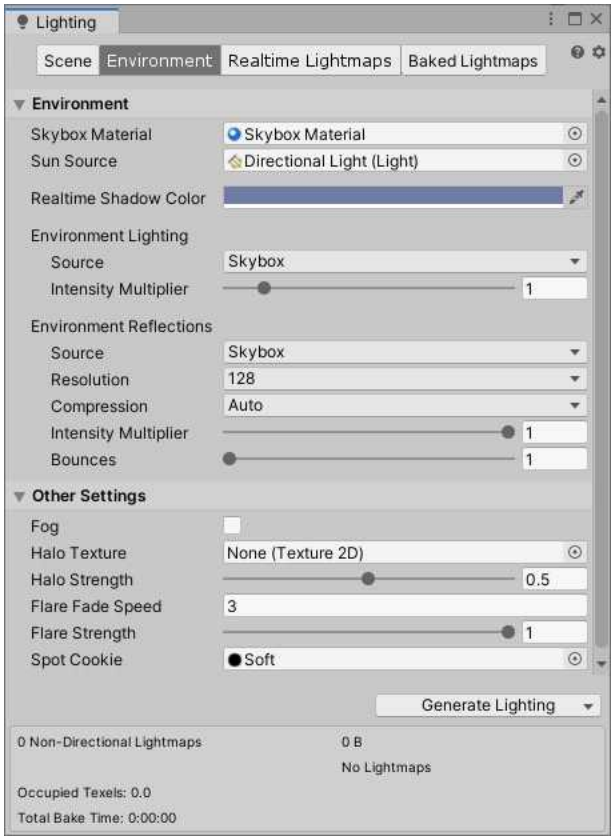
❺ “Import Unity Package” 윈도우에서 “Import”를 선택한다.
임포트가 완료되면 “Assets“ 폴더에 “Models“, “Materials“ 폴더가 생성되고 폴더 안에 여러 가지 애셋들이 있는 것을 확인할 수 있다.

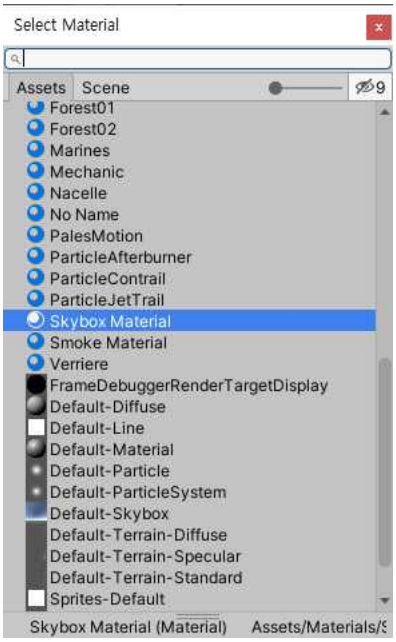


❻ 애셋 폴더에 “Prefabs”, “Scripts” 폴더를 생성한다.

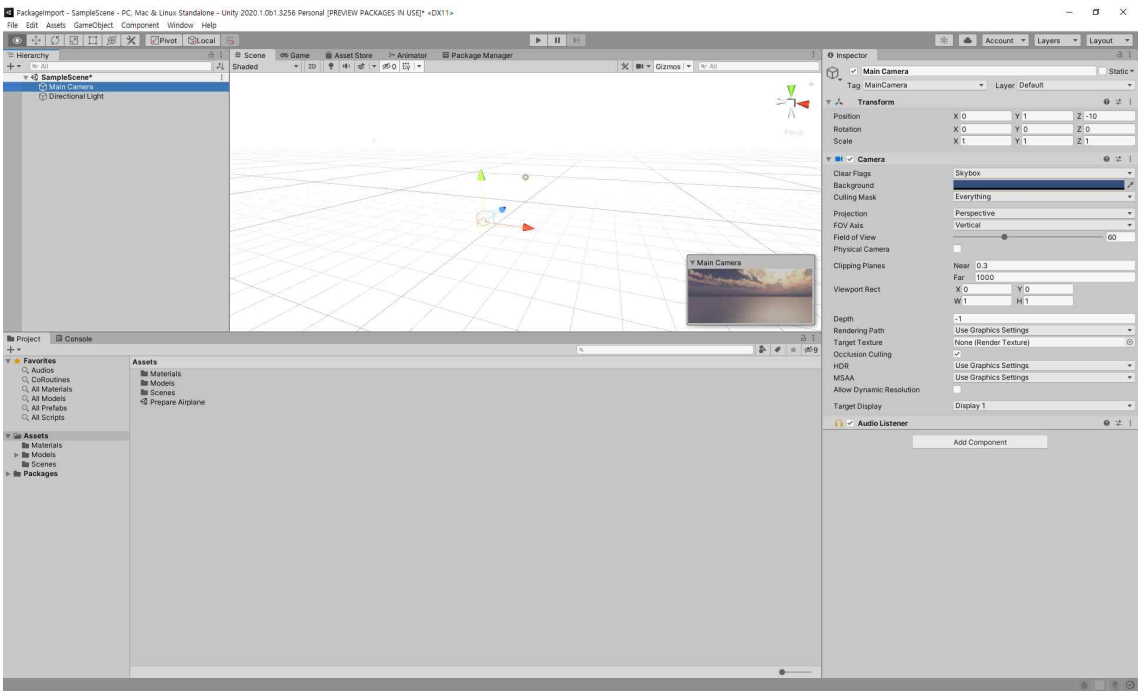


- 스카이 박스(Skybox) 설정하기
  메인 카메라의 배경 스카이 박스를 바꾸자.
  ❶ 메뉴 “Window ▶ Rendering ▶ Lighting“를 선택한다.
  ❷ ”Environment ▶ Skybox Material“의 ”⊙“를 선택한다.
  ❸ ”Select Material“ 윈도우에서 ”Skybox Material“을 선택하고 윈도우를 닫는다.





❹ "Main Camera" 게임 객체를 선택하거나 게임 뷰를 선택하면 스카이 박스가 바뀐 것을 확인할 수 있다.



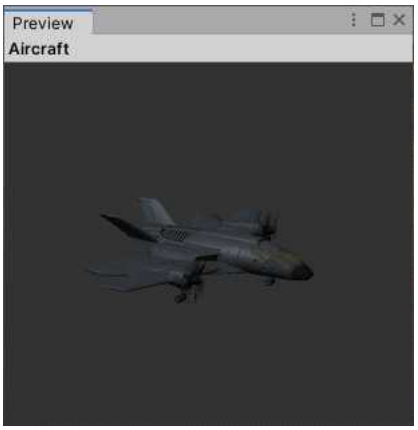
▪ ”Aircraft“ 프리팹 애셋 생성하기

프로젝트 뷰의 “Models ▶ Aircraft ▶ Models“ 폴더에 다음과 같은 3개의 모델이 있다. 이 모델은 비행기를 본체, 날개, 프로펠러 부분으로 분리하여 모델링을 한 것이다. 모델 ”AircraftFuselage“, ”AircraftWingsPropeller“, ”AircraftWingsJet“를 합쳐서 ”Aircraft“ 프리팹 애셋으로 만들자.

❶ 빈 게임 객체(Empty GameObject)를 생성하고 게임 객체의 이름을 “Aircraft”로 변경한다.
“GameObject ▶ Create Empty“

❷ 모델 ”AircraftFuselage“, ”AircraftWingsPropeller“, ”AircraftWingsJet“을 게임 객체 “Aircraft”의 자식 게임 객체로 생성한다.

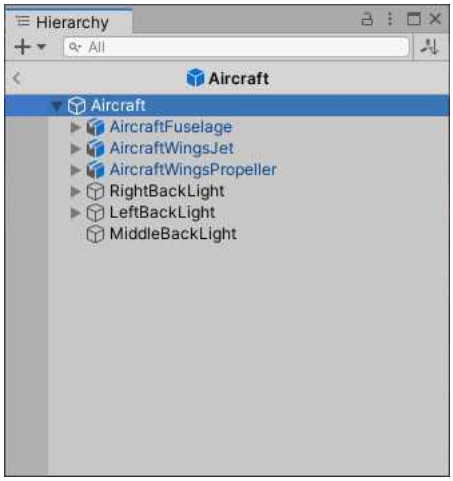
❸ ”AircraftFuselage“, ”AircraftWingsPropeller“, ”AircraftWingsJet“ 게임 객체의 위치와 방향을 조절해서 완성된 비행기 게임 객체를 만든다.



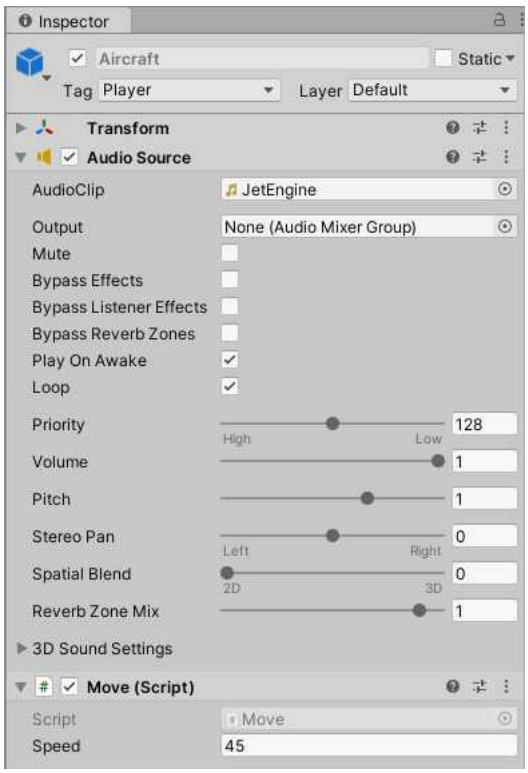
❹ 게임 객체 “Aircraft”를 복제하여 “Player” 게임 객체를 생성한다.

❺ 게임 객체 “Aircraft”를 프리팹 애셋으로 생성한다.

❻ 게임 객체 “Aircraft”를 삭제한다.



- "Aircraft" 프리팹 애셋에 컴포넌트 연결하기
  - ❶ "Scripts" 폴더에 "Move.cs" 스크립트를 생성하여 "Aircraft" 프리팹 애셋에 연결한다. "Move.cs" 스크립트는 "Aircraft" 프리팹 인스턴스를 z-축 방향으로 1초에 "Speed" 속성만큼 이동하기 위한 것이다.

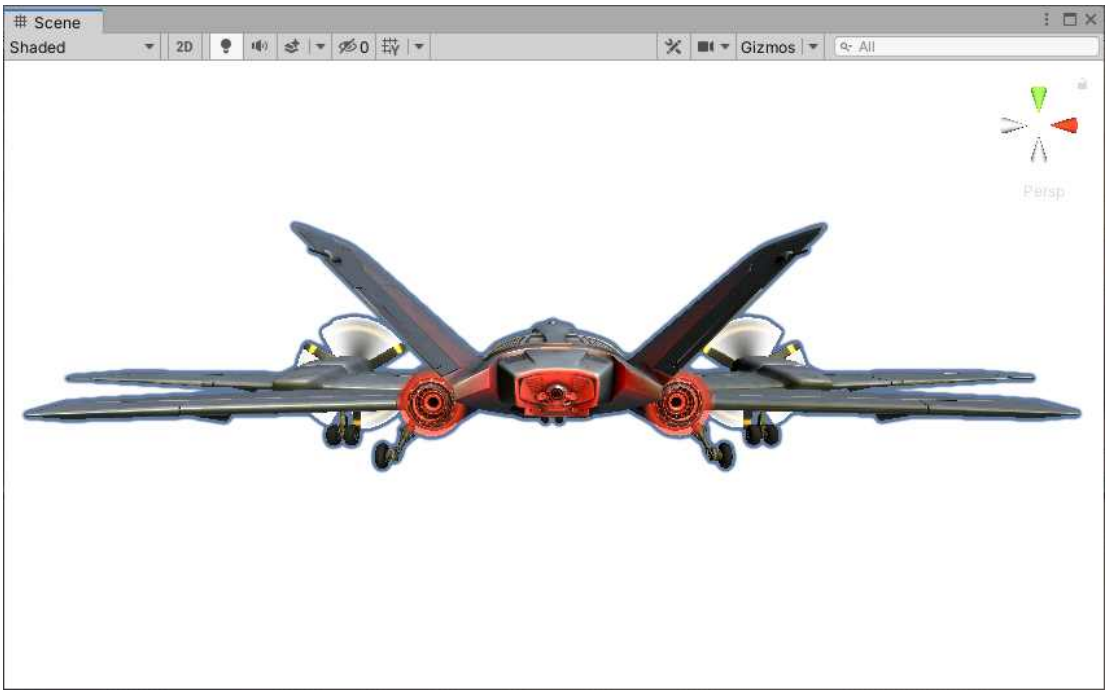



```
public class Move : MonoBehaviour
{
    private float speed = 0.0f;

    void Start()
    {
```

```
        speed = Random.Range(20.0f, 100.0f);
    }

    void Update()
    {
        transform.Translate(0.0f, 0.0f, speed * Time.deltaTime);
    }
}
```

❷ “Audio Source” 컴포넌트를 추가한다.
“Add Component ▶ Audio ▶ Audio Source”
“Audio Source” 컴포넌트는 소리를 재생할 수 있는 컴포넌트이다. “Models ▶ Aircraft ▶ Audio” 폴더의 “JetEngine.wav” 사운드 애셋을 “AudioClip” 속성의 값으로 설정한다.

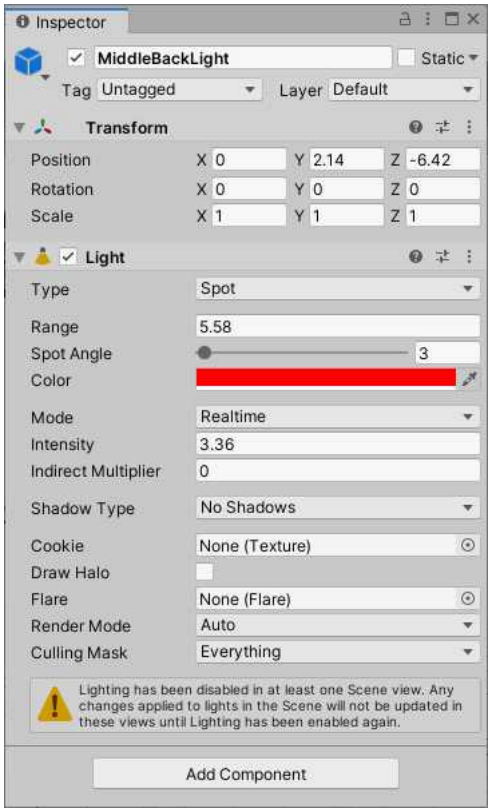
- ”Player“ 게임 객체 설정하기
”Player“ 게임 객체의 뒤 모양이 다음과 같이 보일 수 있도록 설정하자.



❶ ”Player“ 게임 객체의 자식 게임 객체로 조명 게임 객체(Spotlight)를 3개 생성하고 게임 객체의 이름을 ”LeftBackLight“, ”RightBackLight“, ”MiddleBackLight“로 설정한다.
“GameObject ▶ Light ▶ Spotlight“

”MiddleBackLight“ 게임 객체의 위치(Position)는 ”Y: 2.14”, ”Z: -6.42”로 설정한다. ”Light” 컴포넌트에서 “Range: 5.58”, ”Spot Angle: 3”, ”Intensity: 3.36”를

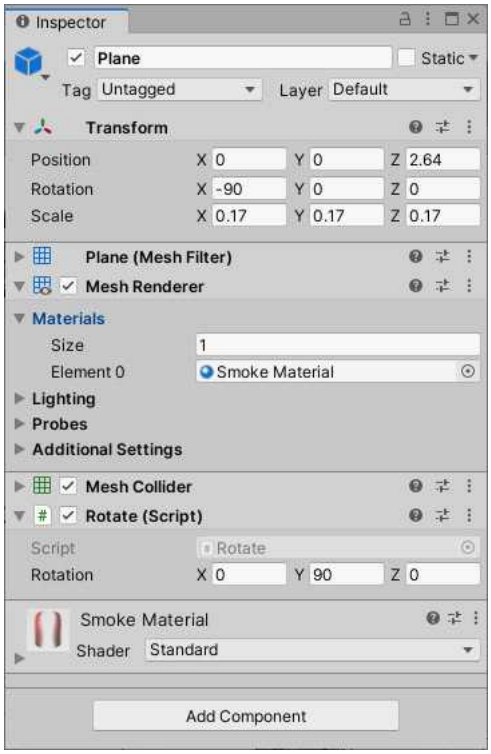
설정한다.
”LeftBackLight“ 게임 객체의 위치는 ”X: -2.56”, ”Y: 1.84”, ”Z: -5.86”으로 설정한다. ”Light” 컴포넌트에서 “Range: 9.4”, ”Spot Angle: 25”, ”Intensity: 5”를 설정한다.
”RightBackLight“ 게임 객체의 위치는 ”X: 2.56”, ”Y: 1.84”, ”Z: -5.86”으로 설정한다. ”Light” 컴포넌트에서 “Range: 9.4”, ”Spot Angle: 25”, ”Intensity: 5”를 설정한다.



❷ ”LeftBackLight“ 게임 객체의 자식 게임 객체로 평면 게임 객체를 생성하고, ”Transform“ 컴포넌트를 다음 그림과 같이 설정한다. ”Mesh Renderer“ 컴포넌트의 ”Materials“ 속성 값으로 ”Assets ▶ Materials“ 폴더의 ”Smoke Material “을 설정한다.

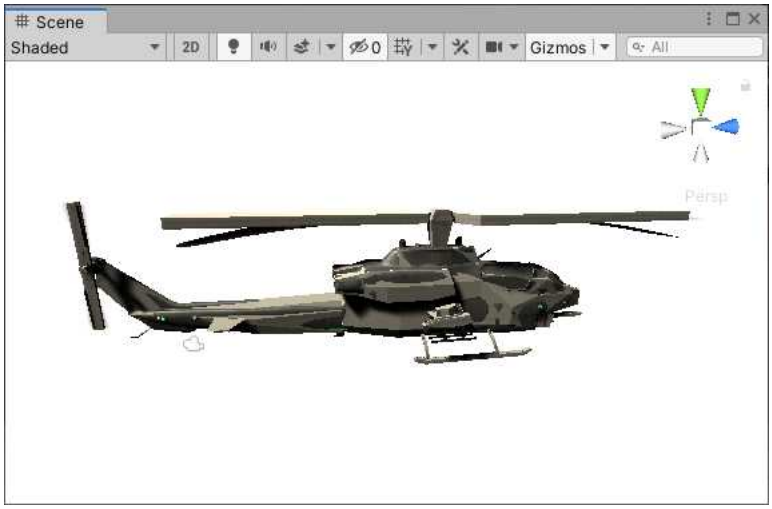
”LeftBackLight“ 게임 객체에 ”Rotate.cs” 스크립트를 연결하고 y-축 회전 값을 90으로 설정한다.

평면 게임 객체를 복제하여 ”RightBackLight“ 게임 객체의 자식 게임 객체로 설정한다.

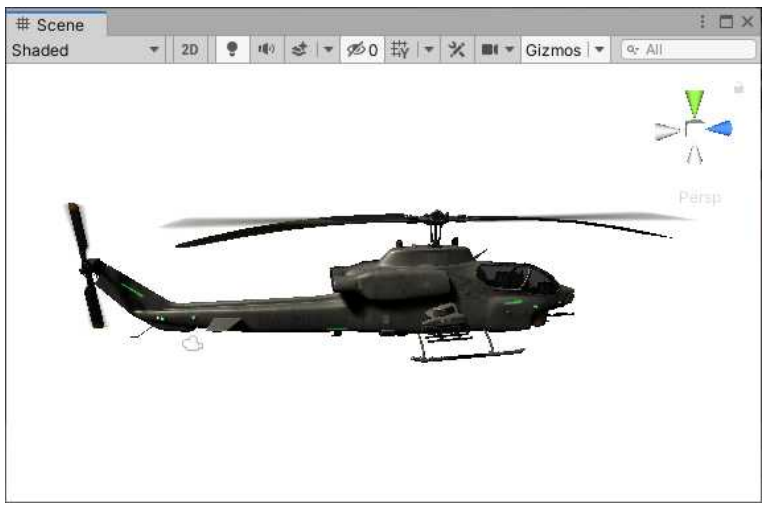


- "SuperCobra" 프리팹 애셋 생성하기
  프로젝트 뷰의 "Models ▶ SuperCobra ▶ Models" 폴더에 다음과 같은 모델이 있다. 이 모델을 "SuperCobra" 프리팹 애셋으로 만들자.

❶ "SuperCobra.fbx" 모델을 게임 객체로 생성한다. 씬 뷰에서 모델을 확인하면 조금 이상하게 보인다. 이것은 계층 구조의 자식 게임 객체들이 충돌 처리를 위하여 만든 충돌 바운딩 박스(Bounding Box)를 가지고 있고 바운딩 박스가 렌더링되기 때문이다.



❷ ”SuperCobra.fbx“ 모델의 계층 구조에서 ”Collider*“ 문자열을 가진 자식 게임 객체들을 모두 제거하거나 비활성화(Inactive)시킨다. 가장 쉬운 방법은 계층 뷰의 검색 창에서 ”Collider“를 입력하여 검색을 하고, 검색 결과를 모두 선택하여 비활성화하는 것이다.



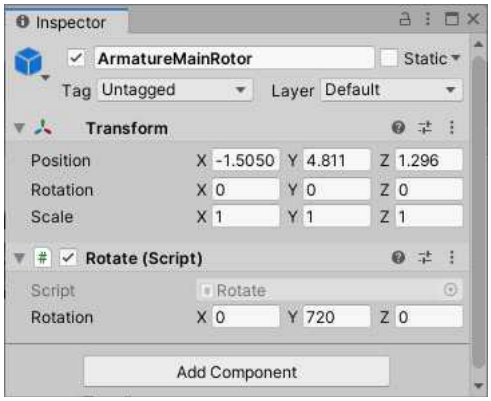
❸ ”SuperCobra“ 게임 객체의 프로펠러를 회전시켜 보자.

프로젝트 뷰의 “Scripts” 폴더에 다음과 같이 “Rotate.cs” 스크립트를 생성하자.

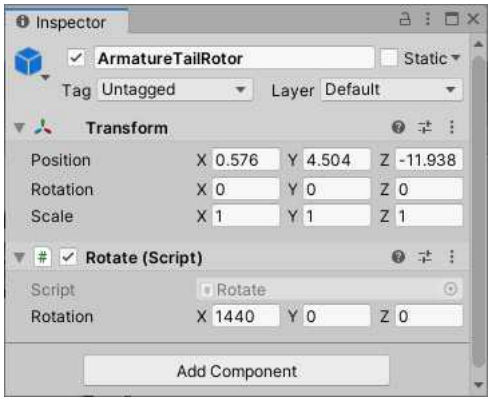
```
public class Rotate : MonoBehaviour
{
    public Vector3 rotation;

    void Update()
    {
        transform.Rotate(rotation * Time.deltaTime);
    }
}
```

“ArmatureMainRotor” 자식 게임 객체에 “Rotate.cs” 스크립트를 연결하고

"Rotation" 속성을 다음과 같이 설정한다. "ArmatureMainRotor" 게임 객체는 y-축이 회전 축이다.



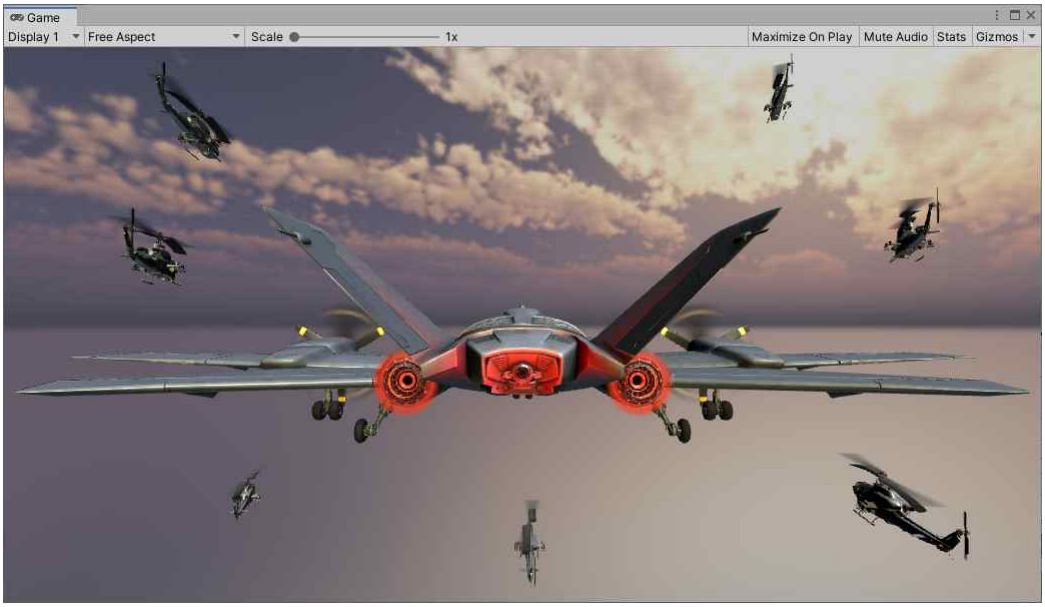
"ArmatureTailRotor" 자식 게임 객체에 "Rotate.cs" 스크립트를 연결하고 "Rotation" 속성을 다음과 같이 설정한다. "ArmatureTailRotor" 게임 객체는 x-축이 회전 축이다.



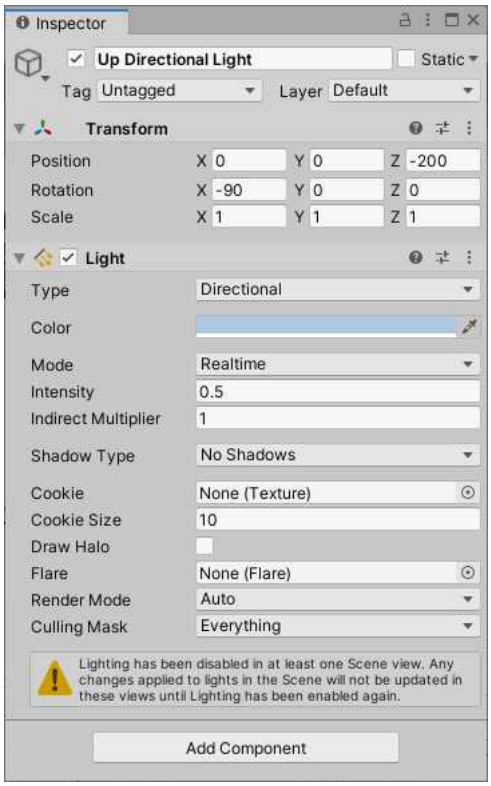
실행해보면 프로펠러가 잘 회전하는 것을 확인할 수 있다.

필요하다면 "Audio Source" 컴포넌트를 추가하고 "Models ▶ Aircraft ▶ Audio" 폴더의 "PropellerEngine.wav" 사운드 애셋을 "AudioClip" 속성의 값으로 설정한다.

❺ 게임 객체 "SuperCobra"를 프리팹 애셋으로 생성한다.
❻ 게임 객체 "SuperCobra"를 적당한 개수만큼 복제하여 다음과 같이 씬에 배치한다.



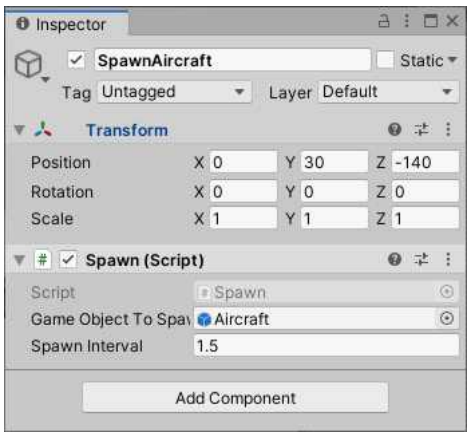
- 조명 생성하기
  방향성 조명 게임 객체를 2개 생성하여 다음과 같이 설정한다.
  "GameObject ▶ Light ▶ Directional Light"

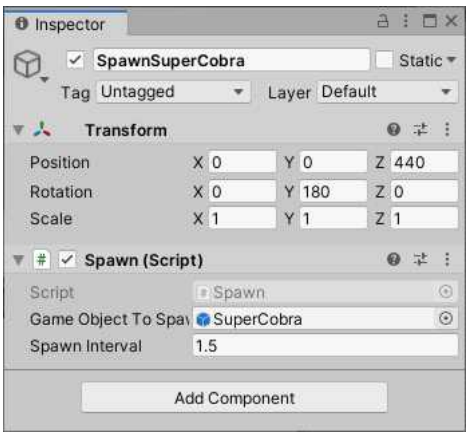


- 스폰 게임 객체 생성하기

  게임 객체를 동적으로 생성하기 위한 두 개의 빈 게임 객체를 생성하고, 게임 객체의 이름을 "SpawnAircraft", "SpawnSuperCobra"로 설정한다.

  "GameObject ▶ Create Empty"



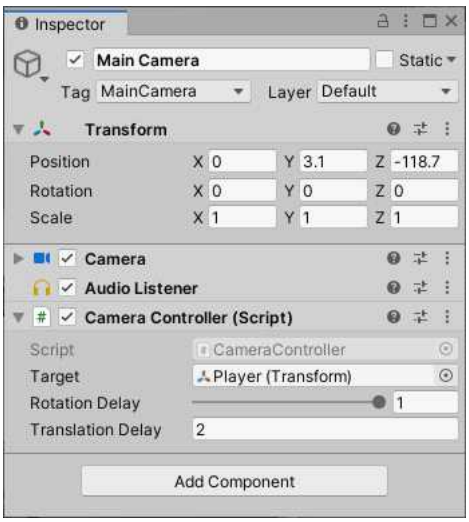


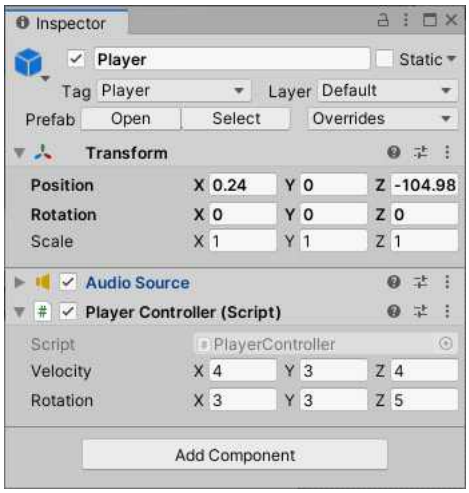
"Spawn.cs" 스크립트를 작성하여 "SpawnAircraft"와 "SpawnSuperCobra" 게임 객체에 연결하고 "Game Object To Spawn"의 값으로 "AirCraft"과 "SuperCobra" 프리팹 애셋을 설정한다.

```
public class Spawn : MonoBehaviour
{
    public GameObject gameObjectToSpawn;
    public float spawnInterval = 1.5f;

    private float lastSpawnTime = 0.0f;

    void Start()
    {
        lastSpawnTime = Time.time;
    }

    void Update()
    {
        if ((Time.time - lastSpawnTime) >= spawnInterval)
        {
            Vector3 pos = new Vector3();
            float random = Random.Range(-1.0f, 1.0f);
            if (random < 0.0f)
            {
                pos.x = transform.position.x + Random.Range(100.0f *
random, -10.0f);
                pos.y = transform.position.y + Random.Range(80.0f *
random, 80.0f);
                pos.z = transform.position.z + Random.Range(-10.0f,
10.0f);
            }
            else if (random > 0.0f)
```

```
            {
                pos.x = transform.position.x + Random.Range(10.0f,
100.0f * random);
                pos.y = transform.position.y + Random.Range(-80.0f,
80.0f * random);
                pos.z = transform.position.z + Random.Range(-10.0f,
10.0f);
            }

            GameObject spawnObject =
Object.Instantiate(gameObjectToSpawn, pos, transform.rotation);
            Move move = spawnObject.GetComponent<Move>();
            if (move != null) move.speed *= Random.Range(1.0f, 1.5f);
            Object.Destroy(spawnObject, 30.0f);

            lastSpawnTime = Time.time;
        }
    }
}
```

- 플레이어의 이동과 회전 구현하기
  플레이어(“Player” 게임 객체)를 이동하고 회전하면 “Main Camera” 게임 객체는 플레이어를 부드럽게 천천히 따라서 이동하고 회전한다.





화살표 키(← → ↑ ↓), "PageUp", "PageDown"를 누르면 플레이어가 왼쪽, 오른쪽, 앞, 뒤, 위쪽, 아래쪽으로 움직인다.
"Alt" 또는 "Ctrl" 키를 누른 상태에서 마우스 왼쪽 버튼을 누르면 플레이어가 x-축, y-축, z-축을 중심으로 회전한다(강의자료 실행 파일을 참고).
“Esc” 키를 누르면 어떤 회전의 상태에서도 월드좌표계에 정렬(초기 방향)이 되도록 되돌린다.

키 입력과 마우스 회전을 위한 입력은 다음 스크립트를 참고한다.

```
void Update()
{
    bool bLeftMouseButton = Input.GetMouseButton(0);

    if (bLeftMouseButton)
    {
        float x = Input.GetAxis("Mouse X");
        float y = Input.GetAxis("Mouse Y");

        bool bLeftCtrl = Input.GetKey(KeyCode.LeftControl);
        bool bLeftAlt = Input.GetKey(KeyCode.LeftAlt);

        ...
    }
    else
    {
        float x = Input.GetAxis("Horizontal") * velocity.x *
Time.deltaTime;
        float z = Input.GetAxis("Vertical") * velocity.z *
Time.deltaTime;
        float y = 0.0f;

        if (Input.GetKey(KeyCode.PageUp)) y = velocity.y *
Time.deltaTime;
        if (Input.GetKey(KeyCode.PageDown)) y = -velocity.y *
Time.deltaTime;
        ...
    }

    if (Input.GetKeyDown(KeyCode.Escape))
    {
        ...
    }
}
```