Satellite Trajectory Path

CTI One Corporation

3679 Enochs Street

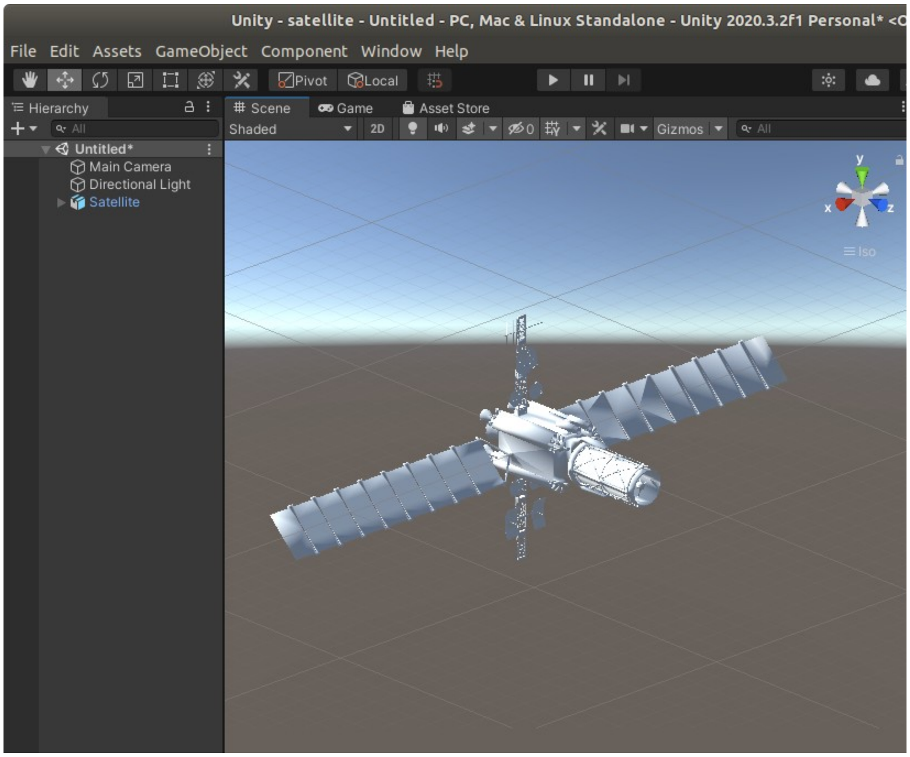
Santa Clara, CA 95051

2021-6-28	Create this document	Chee Vang
2021-7-8	Updated with orbit path, earth texture, background	Chee Vang

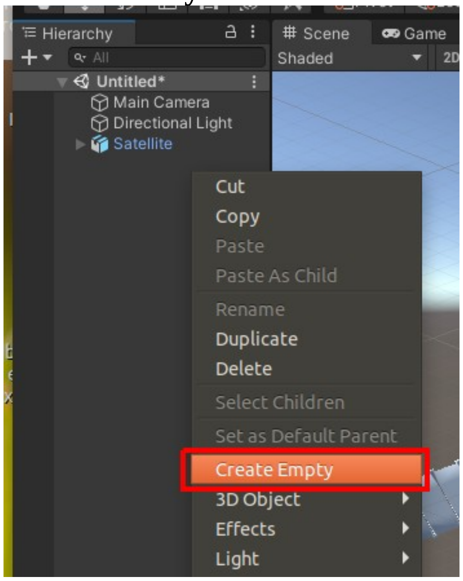
Table of Contents

I. SETTING THE SCENE WITH SATELLITE AND EARTH PLANET

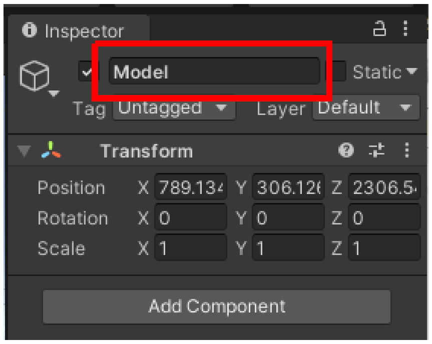
1) Refer to “nnn-n-Import-Satellite-SLDPRT-to-Unity-CV-2021-5-24” to import the Satellite.SLDPRT into Unity. You should have something similar to the image below with only the Satellite, camera, and light in the Hierarchy.


Unity - satellite - Untitled - PC, Mac & Linux Standalone - Unity 2020.3.2f1 Personal* <O
File Edit Assets GameObject Component Window Help
Pivot
Local
Hierarchy
All
Untitled*
Main Camera
Directional Light
Satellite
Scene
Game
Asset Store
Shaded
2D
0
Gizmos
All
y
x
z
Iso

2) Right-click on an empty area in the **Hierarchy** window and select **Create Empty**



3) Select the **GameObject** and rename to “Model” in the **Inspector** window



Model

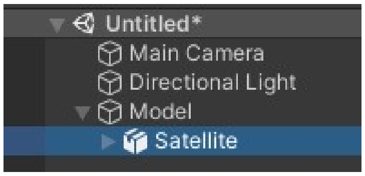
Tag Untagged   Layer Default

Position X 789.134 Y 306.126 Z 2306.54

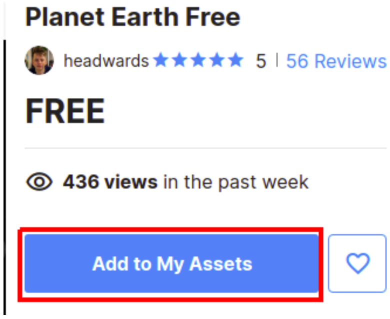
Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

4) Drag and drop **Satellite** in **Model** such that the end results is similar to the image below.



5) Go to https://assetstore.unity.com/packages/3d/environments/sci-fi/planet-earth-free-23399 to use an earth texture from the Asset Store.
5.1) Click on **Add to My Assets** (It will ask for you to log in.)



Planet Earth Free

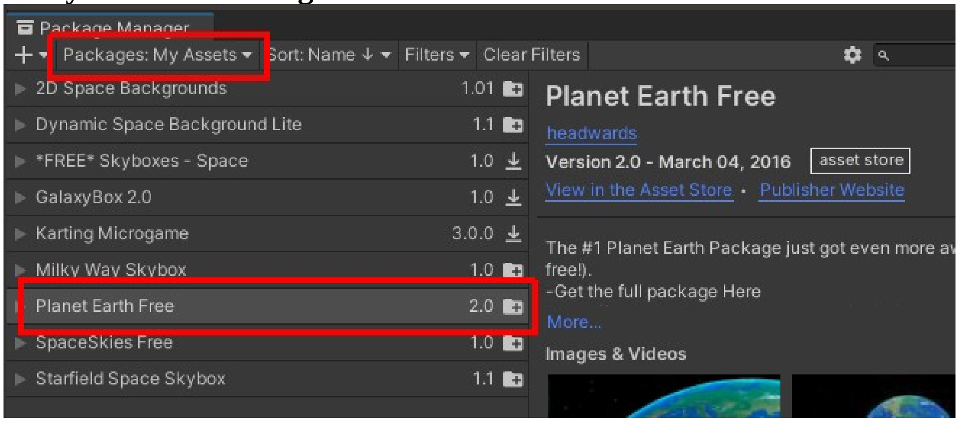
headwards ★★★★★ 5 | 56 Reviews

FREE

436 views in the past week

5.2) Go back to Unity and go to **Window** > **Package Manager**.

5.3) Then, select **My Assets** in **Packages** and select **Planet Earth Free**.



2D Space Backgrounds	1.01
Dynamic Space Background Lite	1.1
FREE Skyboxes - Space	1.0
GalaxyBox 2.0	1.0
Karting Microgame	3.0.0
Milky Way Skybox	1.0
Planet Earth Free	2.0
SpaceSkies Free	1.0
Starfield Space Skybox	1.1

Planet Earth Free

headwards

Version 2.0 - March 04, 2016 asset store

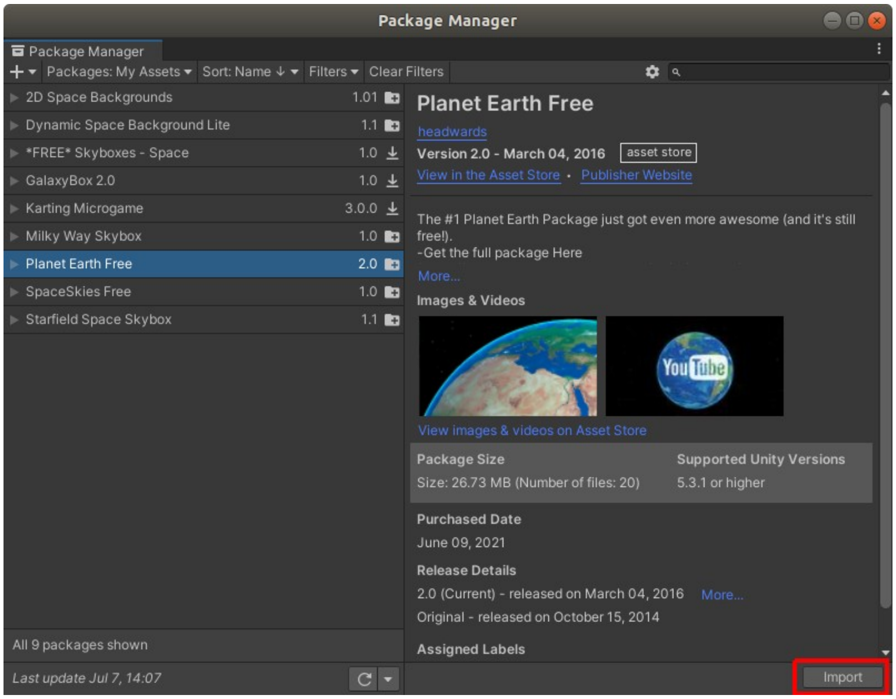
View in the Asset Store • Publisher Website

The #1 Planet Earth Package just got even more av
free!).
-Get the full package Here

More...

Images & Videos

5.4) Click on **Import** in **Package Manager** and click **Import** in **Import Unity Package** for **Planet Earth Free** as shown below.


Package Manager
Package Manager
+ ▾ Packages: My Assets ▾ Sort: Name ↓ ▾ Filters ▾ Clear Filters
Import

2D Space Backgrounds	1.01
Dynamic Space Background Lite	1.1
FREE Skyboxes - Space	1.0
GalaxyBox 2.0	1.0
Karting Microgame	3.0.0
Milky Way Skybox	1.0
Planet Earth Free	2.0
SpaceSkies Free	1.0
Starfield Space Skybox	1.1

All 9 packages shown

Last update Jul 7, 14:07

Planet Earth Free

headwards

Version 2.0 - March 04, 2016 asset store

View in the Asset Store · Publisher Website

The #1 Planet Earth Package just got even more awesome (and it's still free!).
-Get the full package Here

More...

Images & Videos

View images & videos on Asset Store

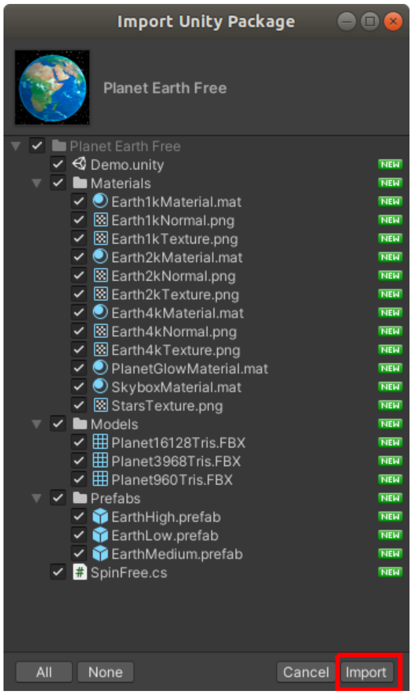
Package Size
Size: 26.73 MB (Number of files: 20)

Supported Unity Versions
5.3.1 or higher

Purchased Date
June 09, 2021

Release Details
2.0 (Current) - released on March 04, 2016 More...
Original - released on October 15, 2014

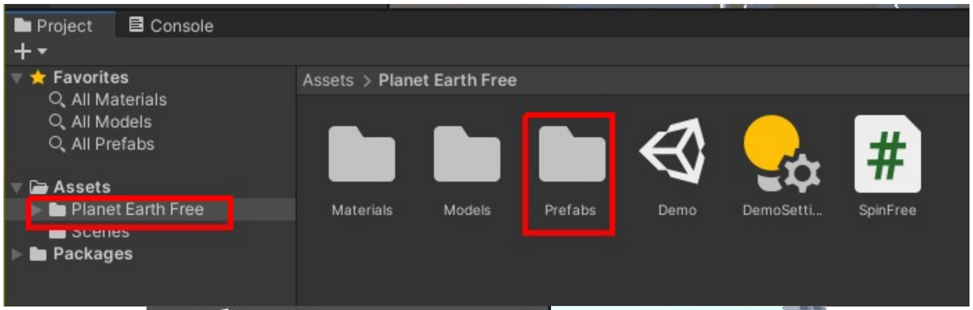
Assigned Labels



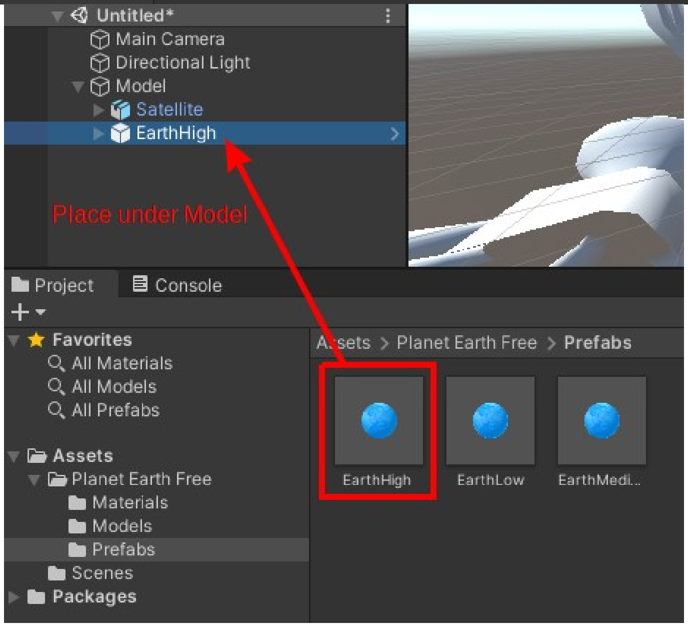
Planet Earth Free

- ✓ Planet Earth Free
  - ✓ Demo.unity NEW
  - ✓ Materials NEW
    - ✓ Earth1kMaterial.mat NEW
    - ✓ Earth1kNormal.png NEW
    - ✓ Earth1kTexture.png NEW
    - ✓ Earth2kMaterial.mat NEW
    - ✓ Earth2kNormal.png NEW
    - ✓ Earth2kTexture.png NEW
    - ✓ Earth4kMaterial.mat NEW
    - ✓ Earth4kNormal.png NEW
    - ✓ Earth4kTexture.png NEW
    - ✓ PlanetGlowMaterial.mat NEW
    - ✓ SkyboxMaterial.mat NEW
    - ✓ StarsTexture.png NEW
  - ✓ Models NEW
    - ✓ Planet16128Tris.FBX NEW
    - ✓ Planet3968Tris.FBX NEW
    - ✓ Planet960Tris.FBX NEW
  - ✓ Prefabs NEW
    - ✓ EarthHigh.prefab NEW
    - ✓ EarthLow.prefab NEW
    - ✓ EarthMedium.prefab NEW
  - ✓ SpinFree.cs NEW

5.5) This asset is located in Assets. Open Prefabs and place it under Model in the Hiearchy.



- Favorites
  - All Materials
  - All Models
  - All Prefabs
- Assets
  - Planet Earth Free
  - Scenes
- Packages



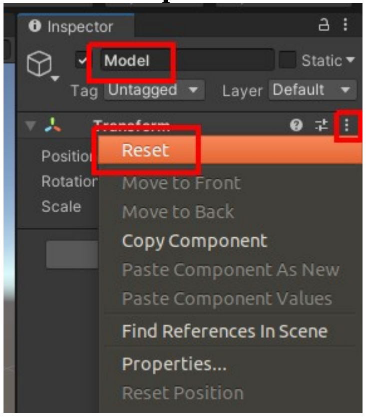
- Untitled*
  - Main Camera
  - Directional Light
  - Model
    - Satellite
    - EarthHigh

Place under Model

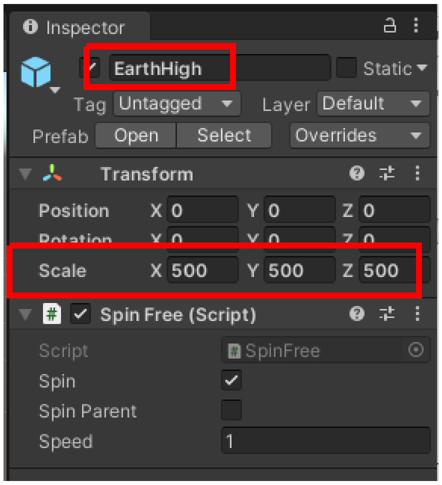
- Favorites
  - All Materials
  - All Models
  - All Prefabs
- Assets
  - Planet Earth Free
    - Materials
    - Models
    - Prefabs
  - Scenes
- Packages

6) Reset the position of **Model** to (0, 0, 0)

6.1) Select **Model** in the **Hierarchy** window
6.2) Select the three dots on **Transform** in the **Inspector** and choose **Reset**

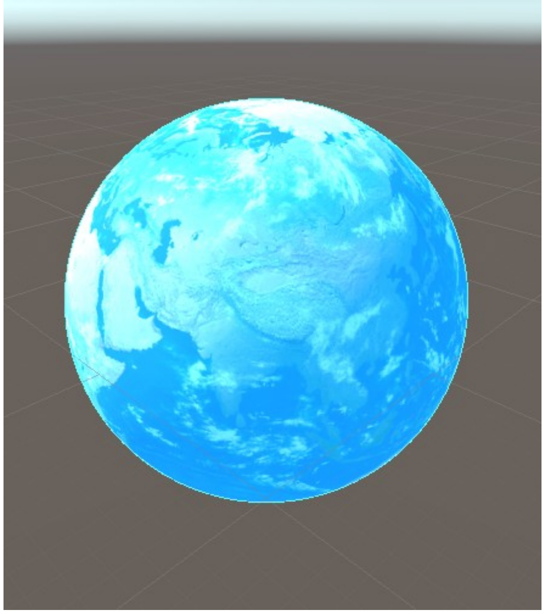


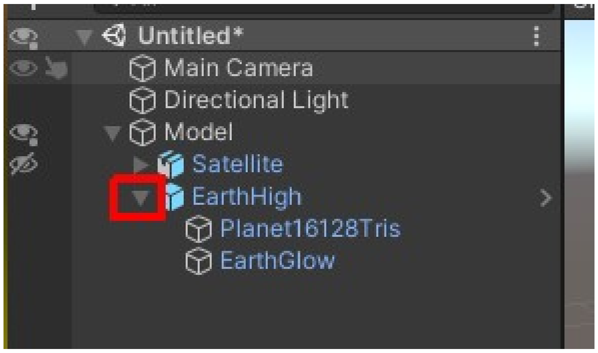
7) Select **EarthHigh** and change the **Scale** to (500, 500, 500)



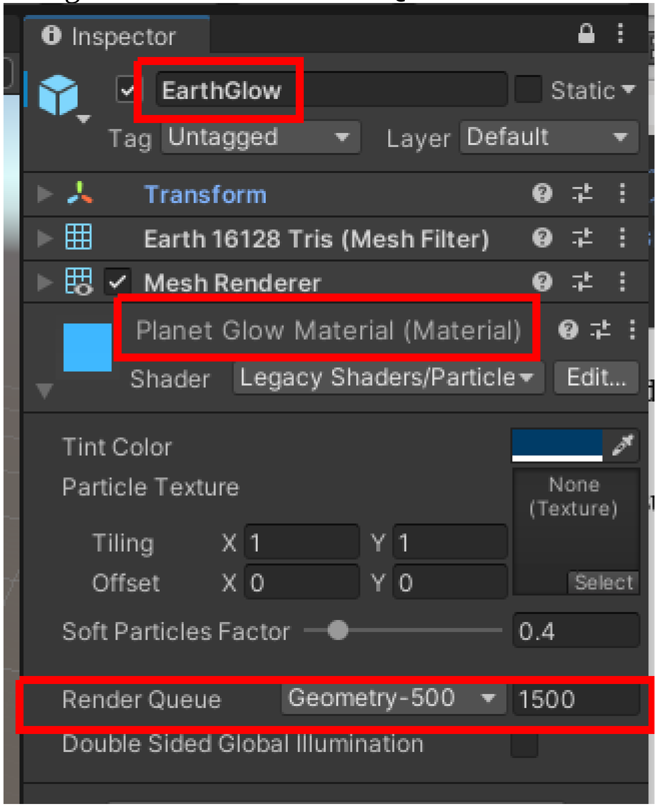
Tag Untagged
Layer Default
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 500 Y 500 Z 500
Script SpinFree
Spin ✓
Spin Parent □
Speed 1

7.1) If **EarthHigh** is all blue (as shown below), click on the drop-down triangle next to **EarthHigh** to reveal its child objects.





7.2) Then select **EarthGlow** to see its components in the **Inspector** window. In **Planet Glow Material**, change the value for **Render Queue** to 1500.



Inspector

☑ EarthGlow □ Static

Tag Untagged Layer Default

Transform

Earth 16128 Tris (Mesh Filter)

☑ Mesh Renderer

Planet Glow Material (Material)

Shader Legacy Shaders/Particle Edit...

Tint Color

Particle Texture None (Texture) Select

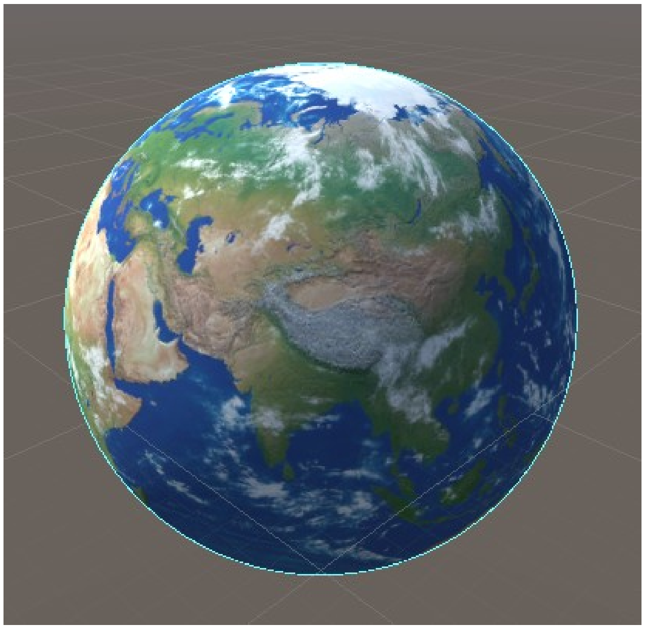
Tiling X 1 Y 1

Offset X 0 Y 0

Soft Particles Factor 0.4

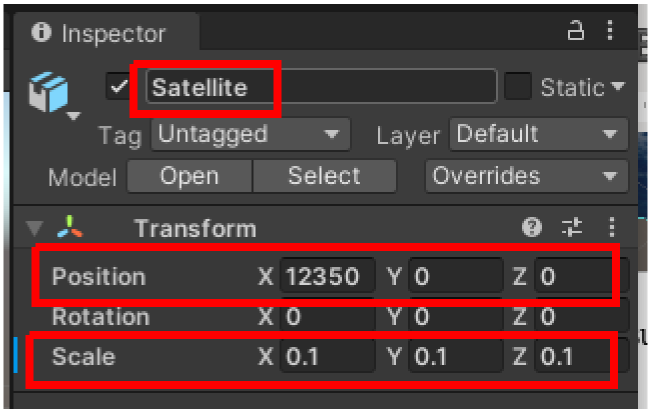
Render Queue Geometry-500 1500

Double Sided Global Illumination □



8) Select **Satellite** in the **Hierarchy** window and change the following such that the satellite is placed outside of the sphere, shown on the right image below.

- **Position** to (12350, 0, 0)
- **Scale** to (0.1, 0.1, 0.1)



Inspector

☑ Satellite □ Static

Tag Untagged Layer Default

Model Open Select Overrides

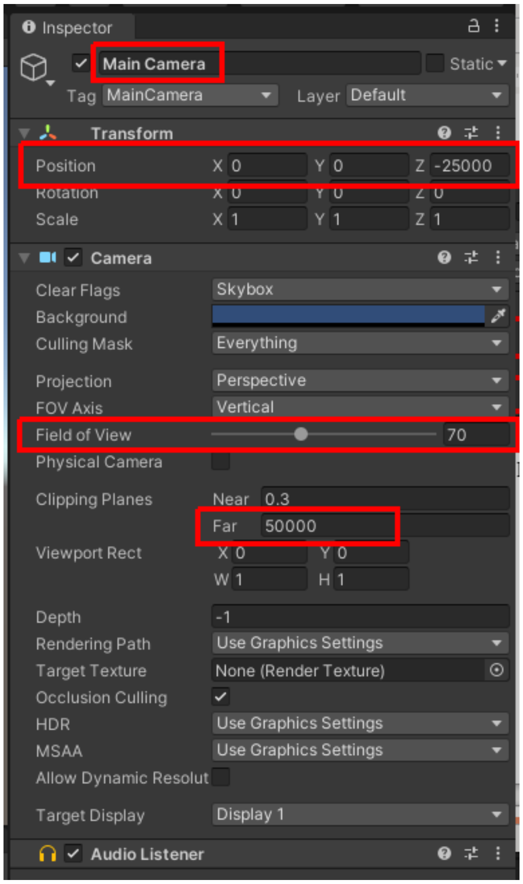
Transform

Position X 12350 Y 0 Z 0

Rotation X 0 Y 0 Z 0

Scale X 0.1 Y 0.1 Z 0.1

9) Change the **Main Camera** view such that the sphere and satellite can be seen in the **Game** view.
9.1) Select Main Camera in the Hierarchy window
9.2) In the **Inspector** window, change **Position** to (0,0,-25000)
9.3) Change **Field of View** to 70
9.4) Change **Clipping Planes** > **Far** to 50000



Inspector

☑ Main Camera Static ▾

Tag MainCamera ▾ Layer Default ▾

Transform

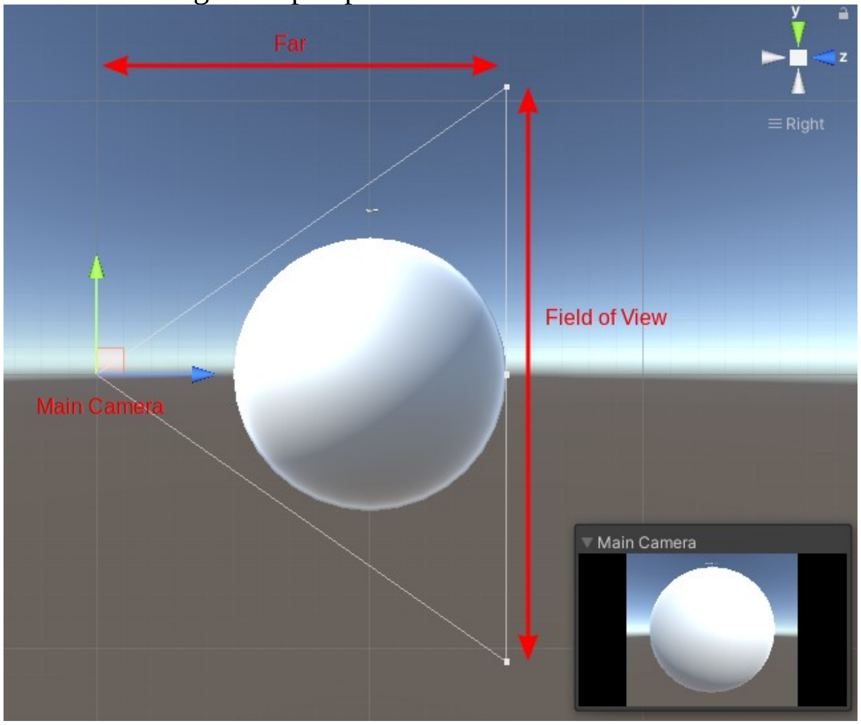
Position	X 0	Y 0	Z -25000
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

☑ Camera

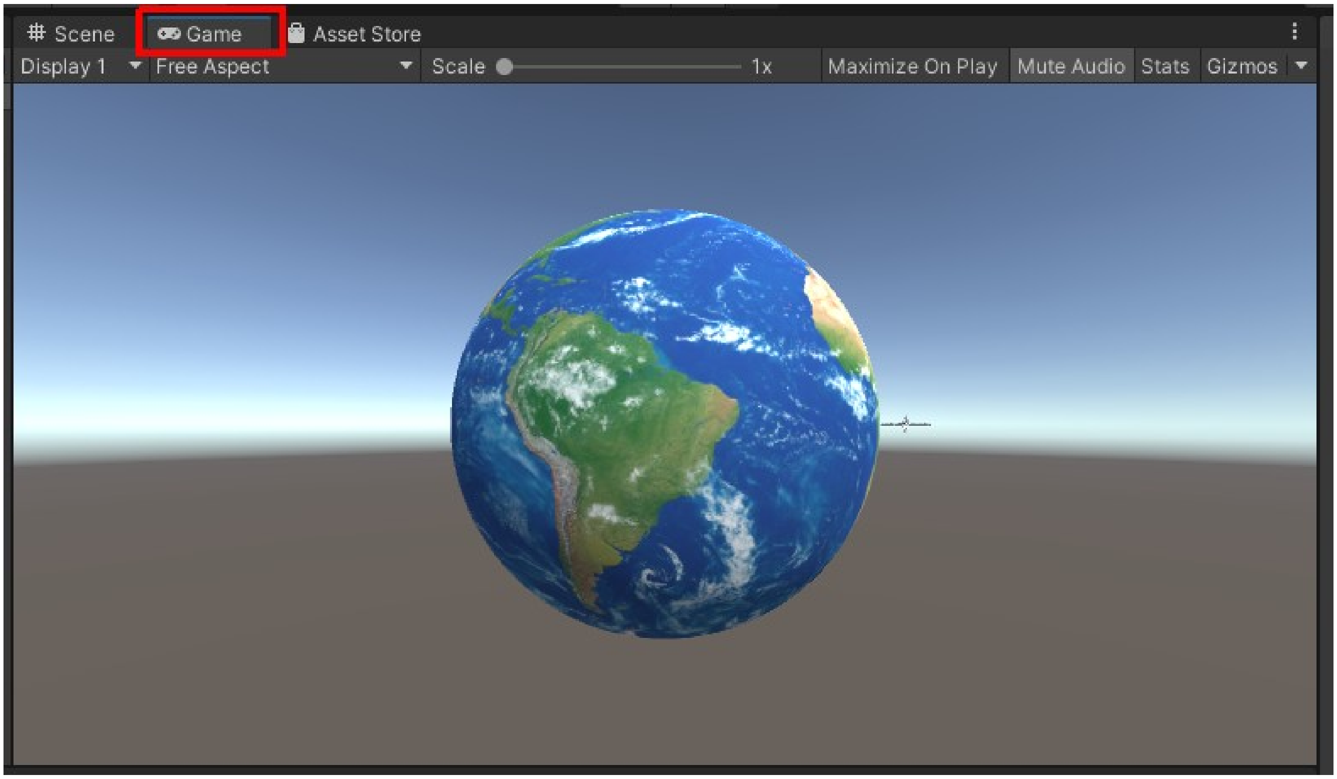
Clear Flags	Skybox
Background	
Culling Mask	Everything
Projection	Perspective
FOV Axis	Vertical
Field of View	70
Physical Camera	□
Clipping Planes	Near 0.3
	Far 50000
Viewport Rect	X 0 Y 0
	W 1 H 1
Depth	-1
Rendering Path	Use Graphics Settings
Target Texture	None (Render Texture)
Occlusion Culling	☑
HDR	Use Graphics Settings
MSAA	Use Graphics Settings
Allow Dynamic Resolut	□
Target Display	Display 1

☑ Audio Listener

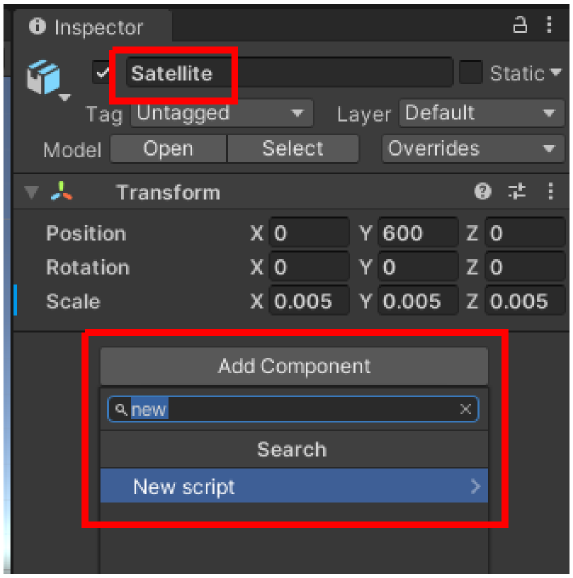
Note: Field of View and Far changes the perspertive of the camera as shown in the image below.



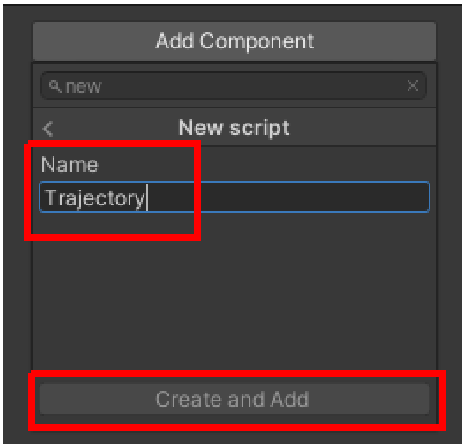
9.5) Go to **Game** view to check if satellite and sphere are visible



10) Create a new folder in **Assets** called “Scripts” by right-clicking in **Assets** and choosing **Create** > **Folder**
11) Add a C# script to Satellite for the trajectory path
11.1) Select **Satellite** and select **Add Component** in the **Inspector** window
11.2) Search and select for **New Script**
11.3) Name the new script “Trajectory”
11.4) Click on **Create and Add**

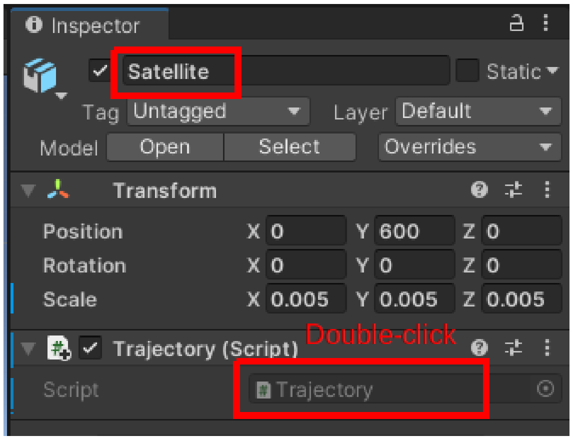


Position X 0 Y 600 Z 0
Rotation X 0 Y 0 Z 0
Scale X 0.005 Y 0.005 Z 0.005
new



Name
Trajectory

11.5) Move **Trajectory.cs** in **Assets** to **Assets/Scripts** to have a clean workspace
12) Double click on **Trajectory** in the **Trajectory (Script)** to open the file



Inspector

Satellite Static

Tag Untagged Layer Default

Model Open Select Overrides

Transform

Position X 0 Y 600 Z 0

Rotation X 0 Y 0 Z 0

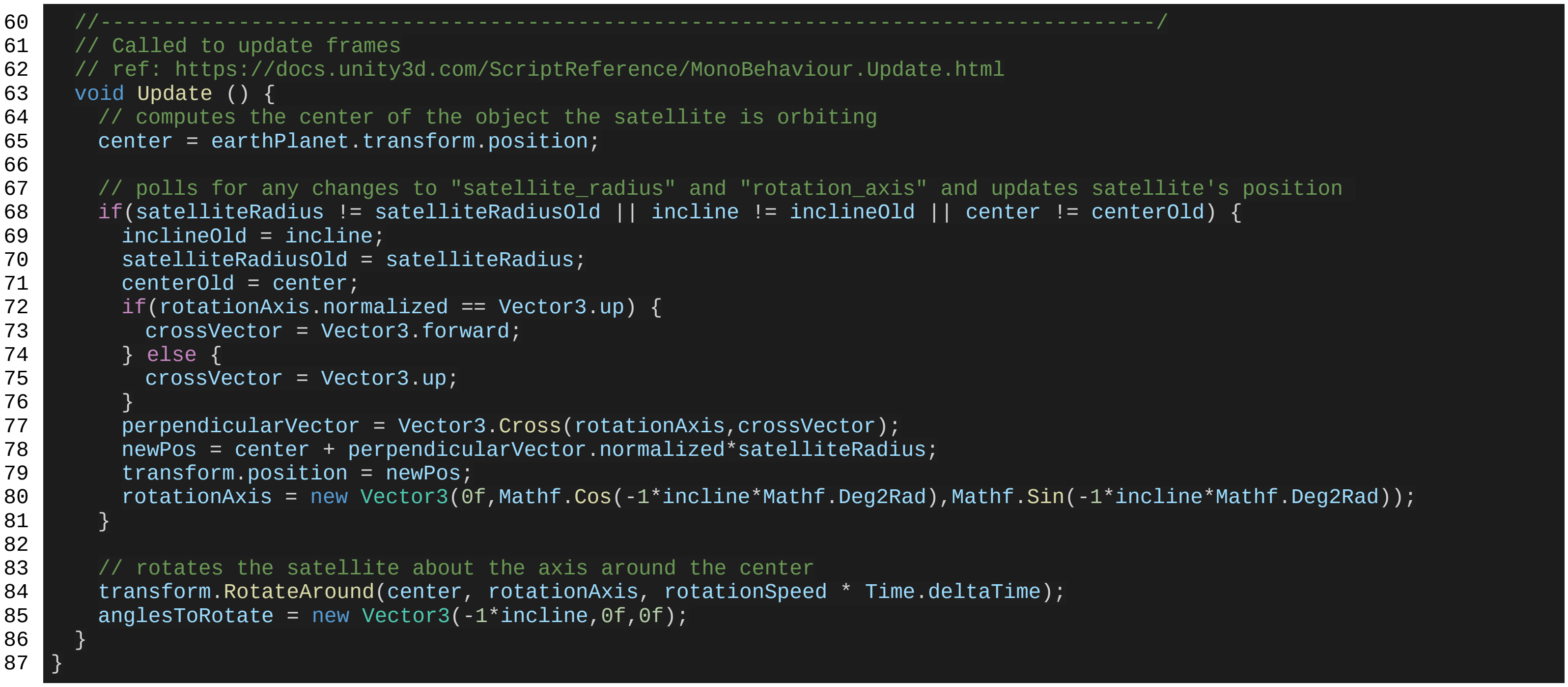
Scale X 0.005 Y 0.005 Z 0.005

Trajectory (Script) Double-click

Script Trajectory

13) Enter the following code to the **Trajectory.cs** file (Note: lines 1-10 includes header information)

```
10   using System.Collections;
11   using System.Collections.Generic;
12   using UnityEngine;
13
14   public class Trajectory : MonoBehaviour
15   {
16   public GameObject earthPlanet; // object that satellite will be orbiting around
17   [HideInInspector] public Vector3 center; // center of both earth and satellite
18   [HideInInspector] public Vector3 rotationAxis = Vector3.forward; // rotates about this axis through the center
19   [Range(11000f, 20000f)]
20   public float satelliteRadius = 12530.0f; // radius of satellite’s orbit between [500,1000] (earthPlanet radius = 500)
21   [Range(1f,360f)]
22   public float rotationSpeed = 20.0f; // 20 degrees/second between [1,360]
23
24   [HideInInspector] public Vector3 perpendicularVector;
25   [HideInInspector] public Vector3 crossVector = Vector3.up; // vector to cross multiply to  perpendicular vector to
     "axis"
26   // https://docs.unity3d.com/ScriptReference/Vector3.html definition of xyz
27   // x-axis = right (1,0,0), y-axis = up (0,1,0), z-axis = forward (0,0,1)
28
29   private Vector3 newPos;
30   private float satelliteRadiusOld; // saves old radius to check if radius changes
31   private float inclineOld; // saves old axis to check if incline changes
32   private Vector3 centerOld; // saves old center to check if center changes
33
34   [Range(-90,90)]
35   public float incline = 0f; // in degrees
36   [HideInInspector] public Vector3 anglesToRotate;
37
38   //---------------------------------------------------------------------------------------/
39   // Used for initialization
40   // ref: https://docs.unity3d.com/ScriptReference/MonoBehaviour.Start.html
41   void Start () {
42     // Sets the frame rate
43     // ref: https://docs.unity3d.com/ScriptReference/Application-targetFrameRate.html
44     //Application.targetFrameRate = 1;
45
46     // sets satellite's initial position to a perpendicular point from "rotation_axis" around "earth_planet"
47     center = earthPlanet.transform.position;
48     rotationAxis = new Vector3(0f,1f,0f);
49     crossVector = Vector3.forward;
50     perpendicularVector = Vector3.Cross(rotationAxis,crossVector);
51     // Vector3.Cross ref: https://docs.unity3d.com/ScriptReference/Vector3.Cross.html
52     transform.position = center + perpendicularVector.normalized*satelliteRadius;
53
54     // initializes satellite_radiusOld and axisOld
55     satelliteRadiusOld = satelliteRadius;
56     inclineOld = incline;
57     centerOld = center;
58   }
59
```

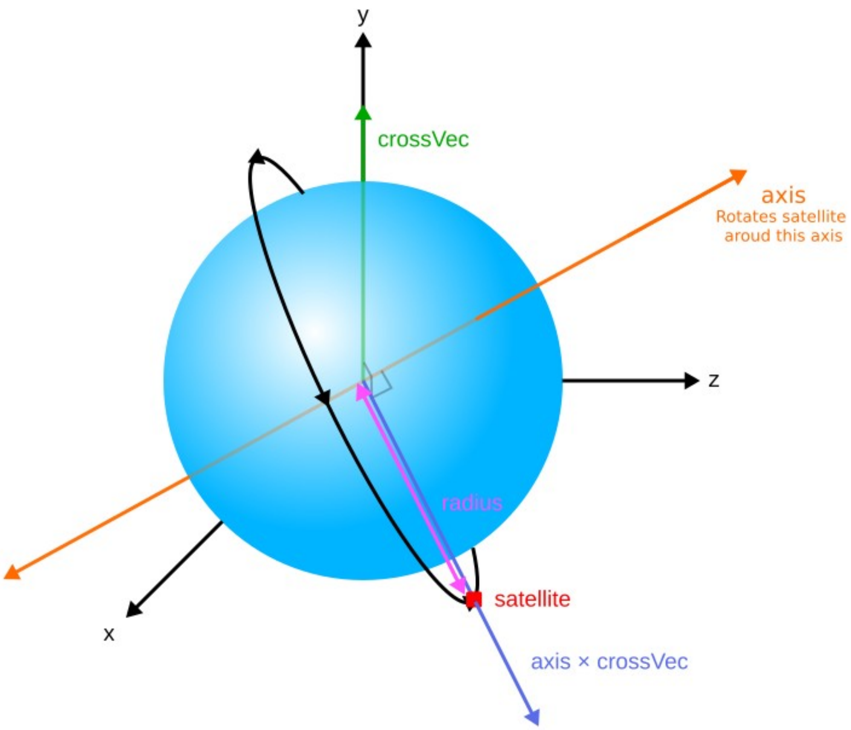


```
60   //-------------------------------------------------------------------------------------/
61   // Called to update frames
62   // ref: https://docs.unity3d.com/ScriptReference/MonoBehaviour.Update.html
63   void Update () {
64     // computes the center of the object the satellite is orbiting
65     center = earthPlanet.transform.position;
66 
67     // polls for any changes to "satellite_radius" and "rotation_axis" and updates satellite's position
68     if(satelliteRadius != satelliteRadiusOld || incline != inclineOld || center != centerOld) {
69       inclineOld = incline;
70       satelliteRadiusOld = satelliteRadius;
71       centerOld = center;
72       if(rotationAxis.normalized == Vector3.up) {
73         crossVector = Vector3.forward;
74       } else {
75         crossVector = Vector3.up;
76       }
77       perpendicularVector = Vector3.Cross(rotationAxis,crossVector);
78       newPos = center + perpendicularVector.normalized*satelliteRadius;
79       transform.position = newPos;
80       rotationAxis = new Vector3(0f,Mathf.Cos(-1*incline*Mathf.Deg2Rad),Mathf.Sin(-1*incline*Mathf.Deg2Rad));
81     }
82 
83     // rotates the satellite about the axis around the center
84     transform.RotateAround(center, rotationAxis, rotationSpeed * Time.deltaTime);
85     anglesToRotate = new Vector3(-1*incline,0f,0f);
86   }
87 }
```

Breakdown of the code:

Lines 41-58: Start() is used for initialization in Unity

- ref: https://docs.unity3d.com/ScriptReference/MonoBehaviour.Start.html
- Line 47-52: initializes satellite’s position to a perpendicular point of axis that is radius far away from the center

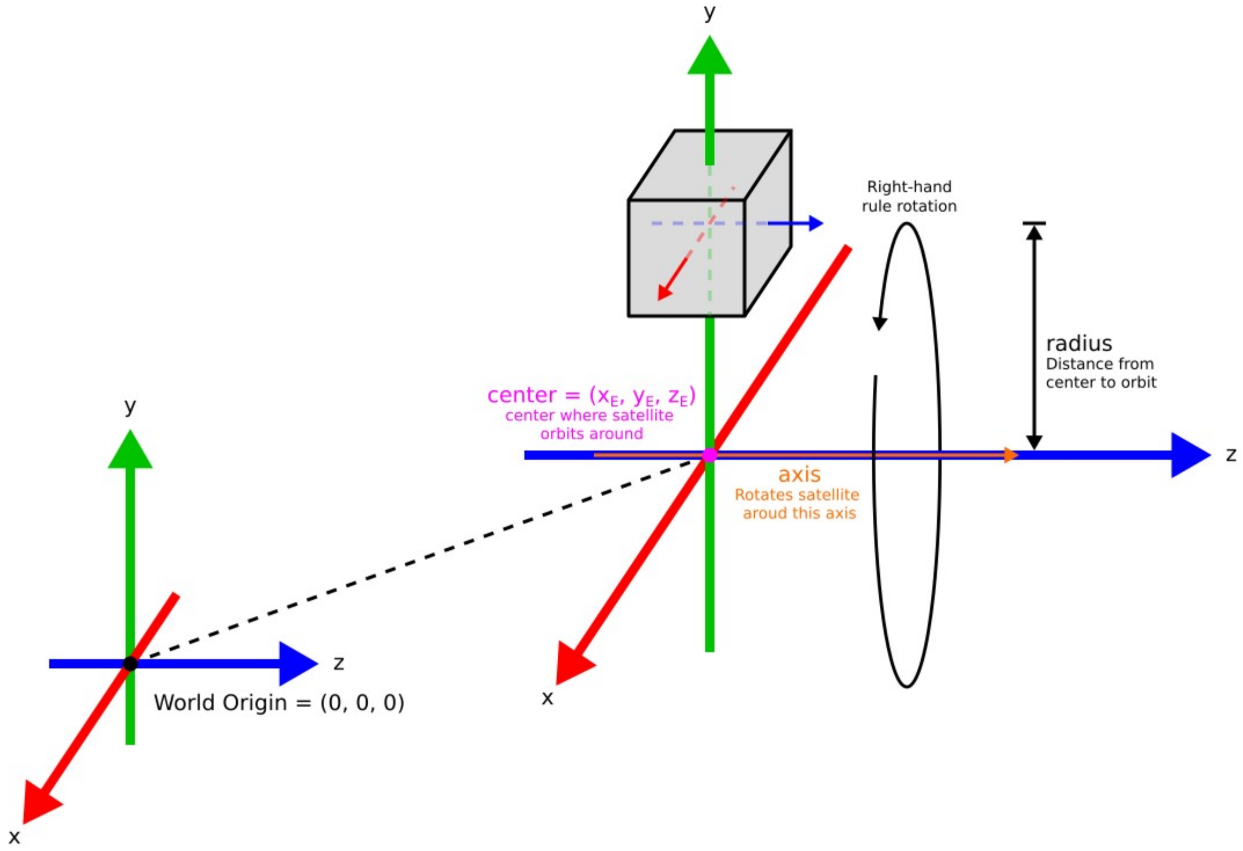


- Line 55-57: initializes variables to check for any changes to radius, axis, and center

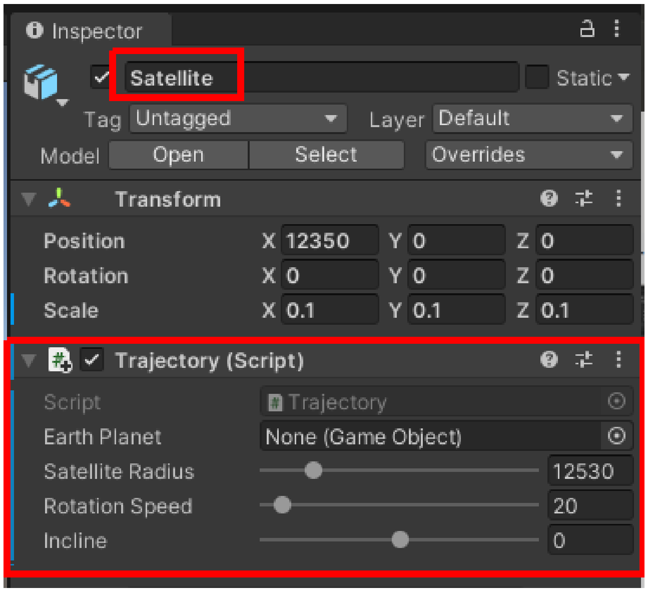
Lines 63-87: Update() is called at every frames to update the scene

- ref: https://docs.unity3d.com/ScriptReference/MonoBehaviour.Update.html

- Line 65: computes the center of earthPlanet
  - if “earth” is moved around, the satellite continues to rotate around it
- Line 68-82: updates the position of the satellite if the radius, axis, incline, or center was changed
- Line 85: rotates satellite around a center around axis
  - RotateAround ref: https://docs.unity3d.com/ScriptReference/Transform.RotateAround.html
  - Syntax: `RotateAround(Vector3 point, Vector3 axis, float angle);`
  - Rotates the transform (satellite) about `axis` passing through `point` in world coordinates by `angle` degrees
- Line 86: updates anglesToRotate variable that is used to render orbital path



14) Save the file and go back to Unity Editor such that the components for **Satellite** are updated in the **Inspector** window



Inspector

☑ Satellite □ Static

Tag Untagged Layer Default

Model Open Select Overrides

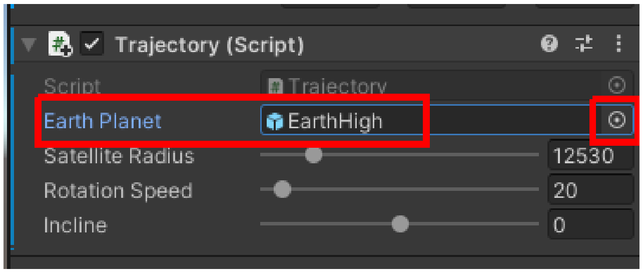
Transform

Position	X 12350	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 0.1	Y 0.1	Z 0.1

☑ Trajectory (Script)

Script	Trajectory
Earth Planet	None (Game Object)
Satellite Radius	12530
Rotation Speed	20
Incline	0

15) Click on the circle (⊙) in **Earth Planet** and select **EarthHigh** to specify the object the satellite will orbit around

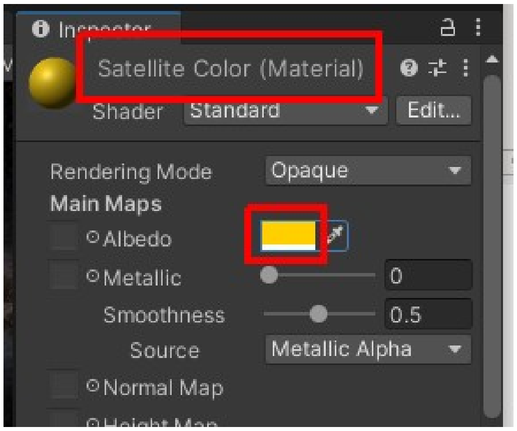


☑ Trajectory (Script)

Script	Trajectory
Earth Planet	EarthHigh
Satellite Radius	12530
Rotation Speed	20
Incline	0

II. MODIFY SATELLITE COLOR

1) Create a folder called “Materials” in **Assets**
2) In the **Materials** folder, right-click and select **Create** > **Material**
3) Rename the material to **Satellite Color**
4) Select **Satellite Colo**r to show it’s properties in the **Inspector** window
5) Click on the color in **Albedo**, select a yellow color



Shader Standard Edit...

Rendering Mode Opaque

Main Maps

Albedo

Metallic 0

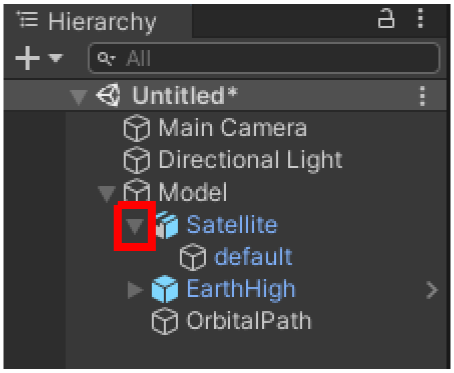
Smoothness 0.5

Source Metallic Alpha

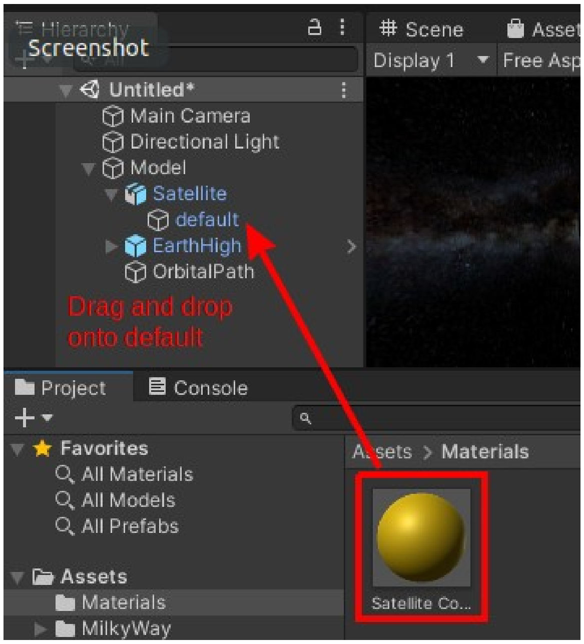
Normal Map

Height Map

6) Click on the triangle next to **Satellite** in the **Hierarchy**

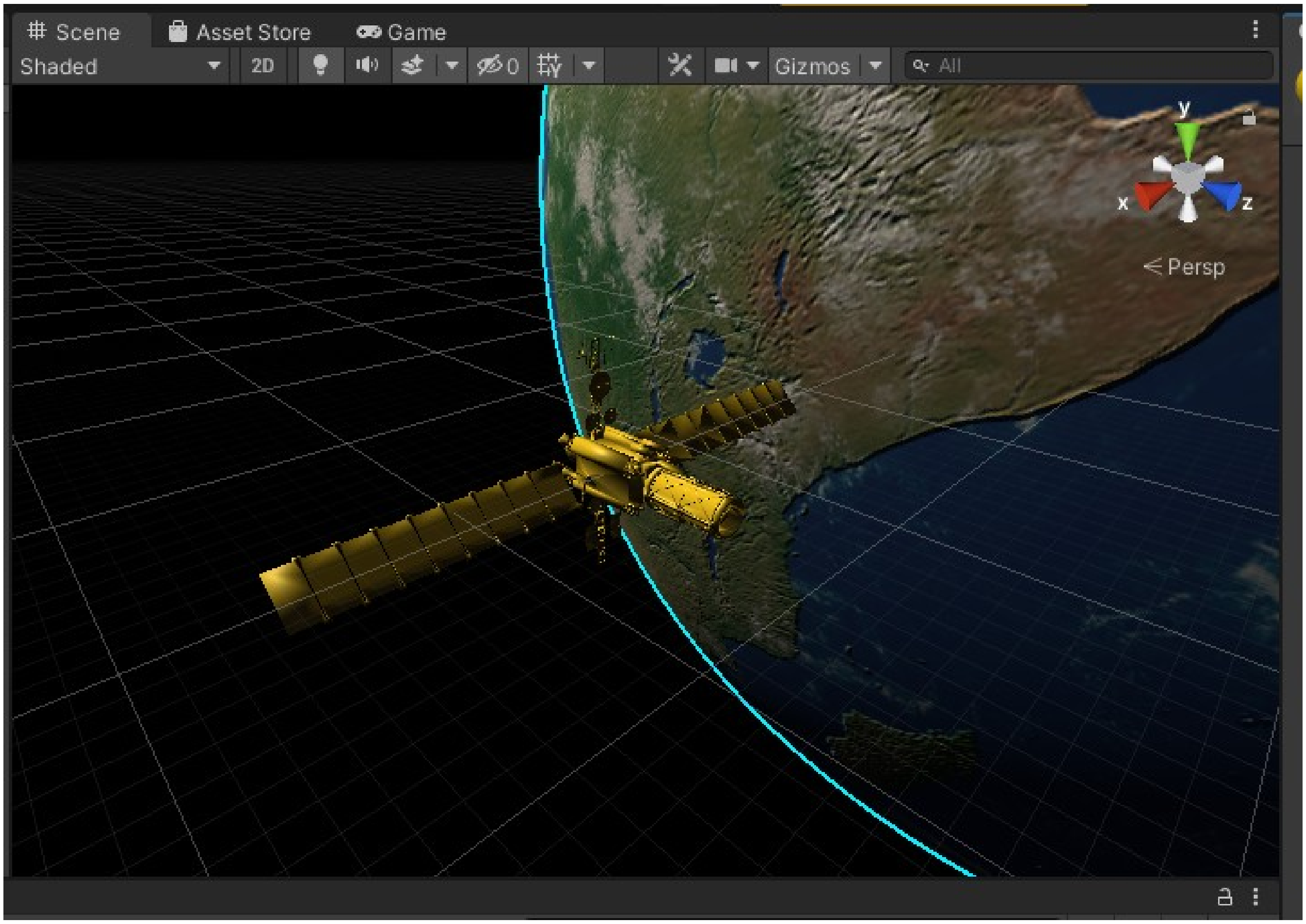


7) Drag and drop **Satellite Color** onto **default**

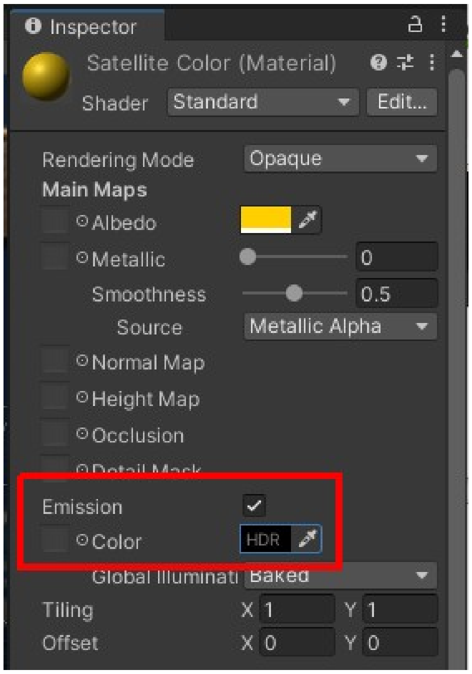


Drag and drop onto default

8) Double-click Satellite to zoom into this object like shown below (the satellite is still too dark)



9) Select the Satellite Color in the Materials folder
10) Check the box next to Emission



Inspector

Satellite Color (Material)

Shader Standard Edit...

Rendering Mode Opaque

Main Maps

Albedo

Metallic 0

Smoothness 0.5

Source Metallic Alpha

Normal Map

Height Map

Occlusion

Detail Mask

Emission ☑

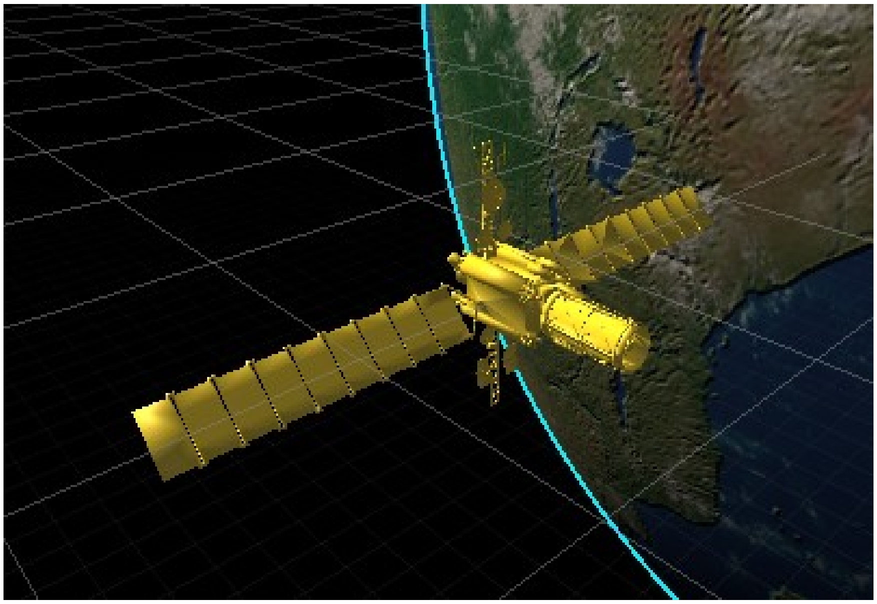
Color HDR

Global Illuminati Baked

Tiling X 1 Y 1

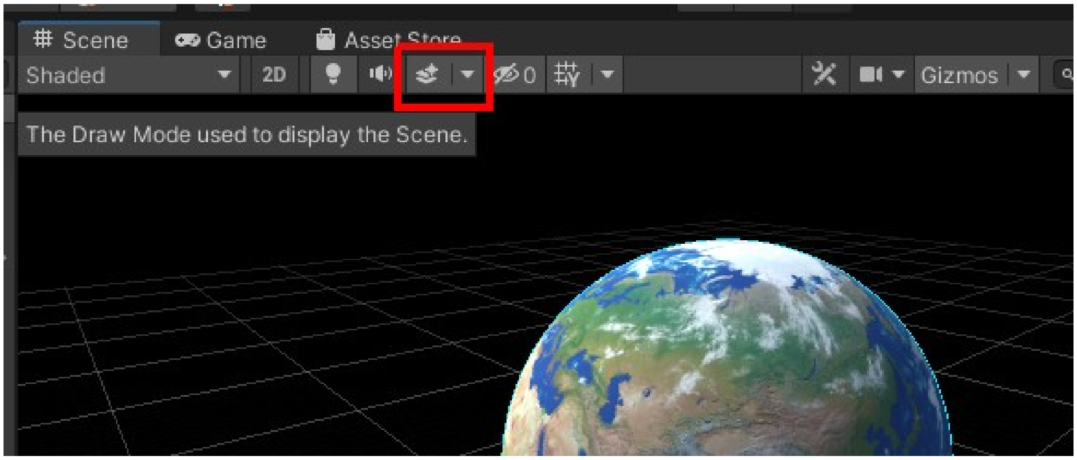
Offset X 0 Y 0

11) Select a color that will brighten the satellite but not washout the details
For example: Albedo RGB = (255, 210, 0) with Emission RGB = (67, 61, 21) gives:



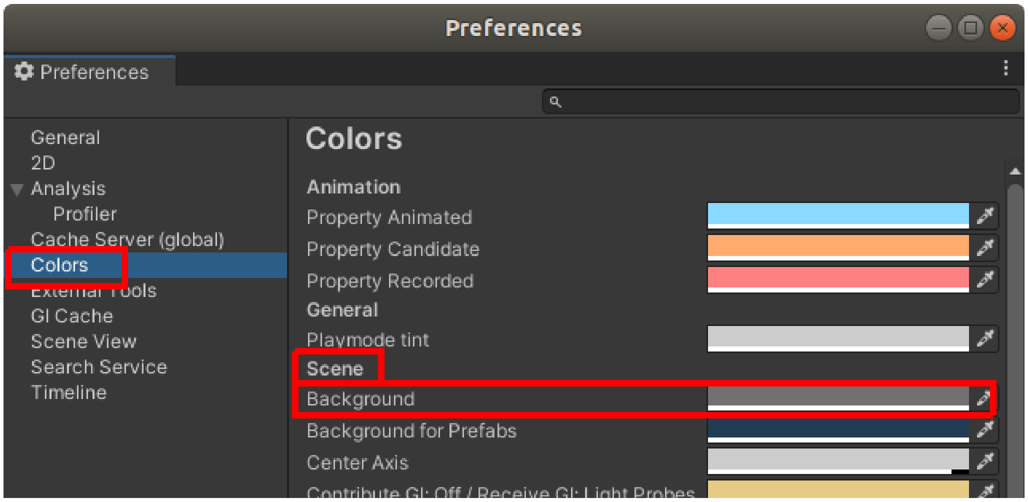
III. CHANGING THE BACKGROUND

To change the background in
1) **Scene view**
1.1) Click on down arrow (▼) next to **Toggle skybox, fog, and various other effects**

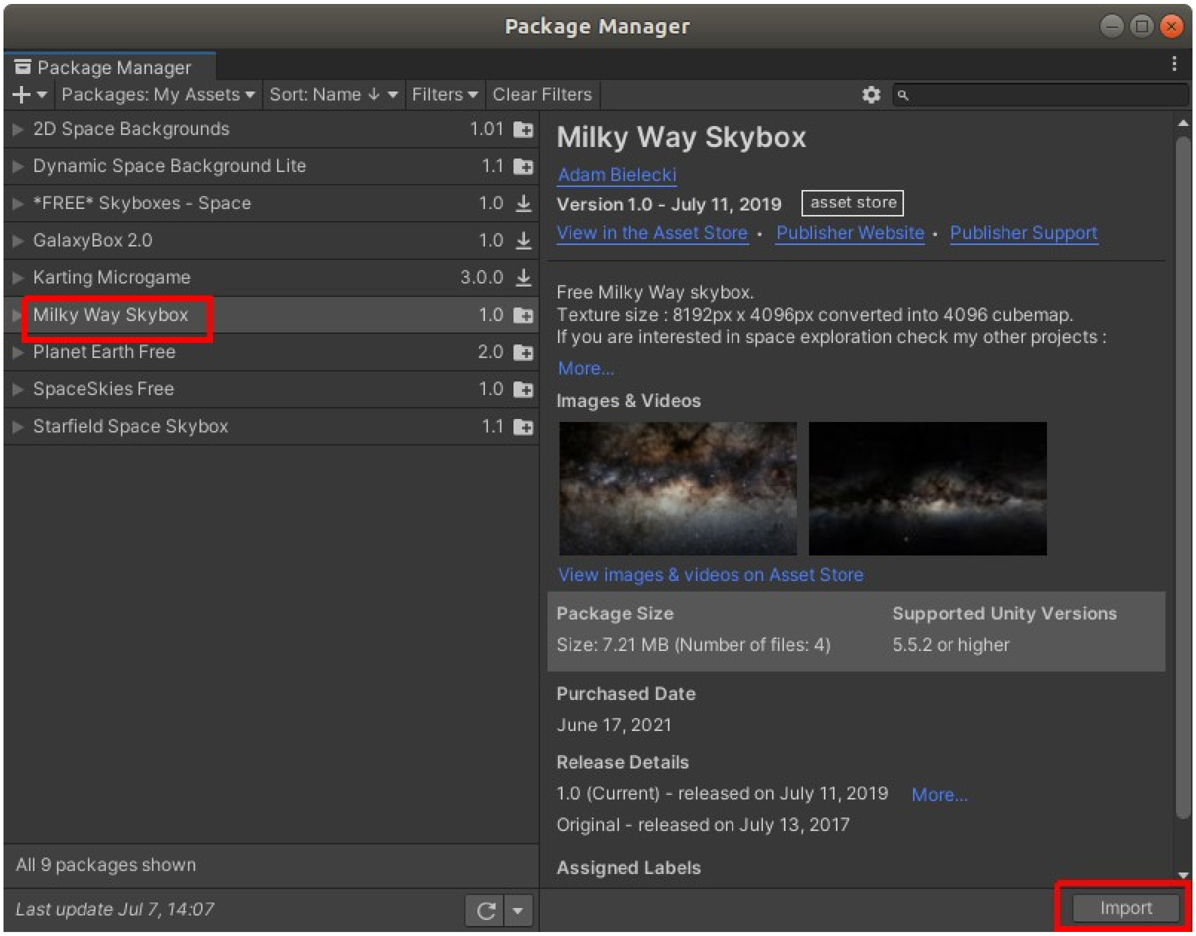


The Draw Mode used to display the Scene.

1.2) De-select **Skybox**
1.3) Then, go to **Edit** > **Perferences** > **Colors** > **Scene** and click on the color for **Background** to choose a dark color for outerspace (i.e. black)



2) **Game view**
2.1) Go to https://assetstore.unity.com/packages/2d/textures-materials/milky-way-skybox-94001 and click on **Add to My Assets**
2.2) Go back to Unity and go to **Window** > **Package Manager**
2.3) Change **Packages** to **My Assets** and import **Milky Way Skybox**



Free Milky Way skybox.
Texture size : 8192px x 4096px converted into 4096 cubemap.
If you are interested in space exploration check my other projects :

Package Size
Size: 7.21 MB (Number of files: 4)

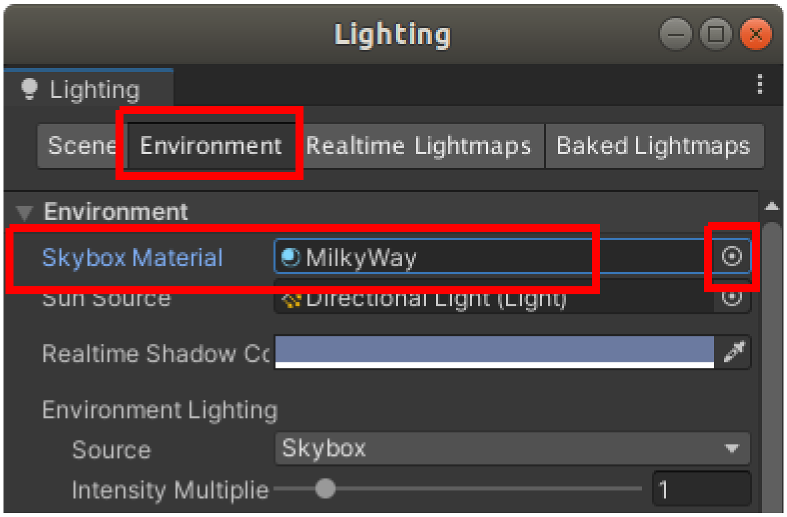
Supported Unity Versions
5.5.2 or higher

Purchased Date
June 17, 2021

Release Details
1.0 (Current) - released on July 11, 2019
Original - released on July 13, 2017

Assigned Labels

2.4) Then, go to **Window** > **Rendering** > **Lighting** > **Environment**
2.5) Click on the circle (⊙) in **Skybox Material** and select **MilkyWay**



Environment

Skybox Material MilkyWay

Sun Source Directional Light (Light)

Realtime Shadow Cc

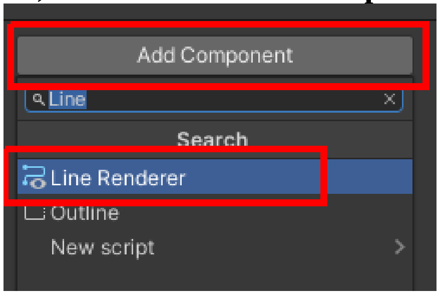
Environment Lighting

Source Skybox

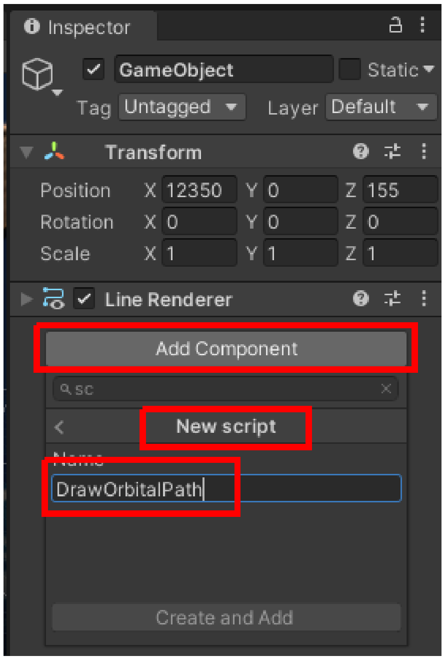
Intensity Multiplie 1

IV. RENDER THE ORBITAL PATH

1) Right click **Model** and select **Create Empty**
2) Rename the new object **OrbitalPath**
3) While **OrbitalPath** is still selected, click on **Add New Component** and add **Line Renderer**



4) Add a C# Script to **OrbitalPath** by clicking on **Add New Coponent**



GameObject Static

Tag Untagged Layer Default

Position X 12350 Y 0 Z 155

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Name

DrawOrbitalPath

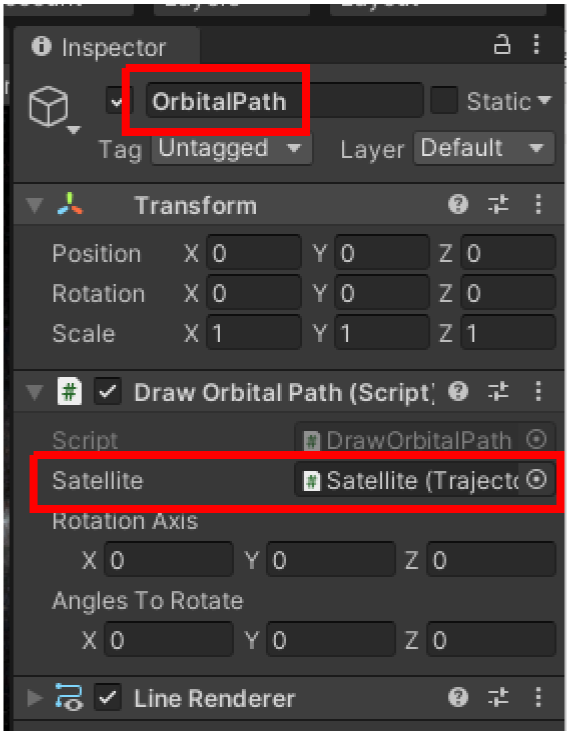
5) Name the file to **DrawOrbitalPath** and place it into the **Scripts** folder

6) Enter the following code into the file

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DrawOrbitalPath : MonoBehaviour
{
  public Trajectory satellite;
  LineRenderer trajPath;
  private int segments = 360;

  //public float incline = 0f;
  public Vector3 rotationAxis;
  public Vector3 anglesToRotate = new Vector3(0f,0f,0f);

  // Start is called before the first frame update
  void Start() {
  // Debug.LogWarning(satellite.satelliteRadius);
    trajPath = gameObject.GetComponent<LineRenderer>();
    trajPath.useWorldSpace = false;
    trajPath.material = new Material(Shader.Find("Unlit/Color"));
    trajPath.material.color = Color.red;
    trajPath.startWidth = 50f;
    trajPath.endWidth = 50f;
    trajPath.positionCount = segments + 1;
  }

  // Update is called once per frame
  void Update() {
    drawCircle(satellite.satelliteRadius);
    rotateTrajPath();
  }

  // creates 360 segments to render the orbital path using x,y,z values
  void drawCircle(float radius) {
    float x;
    float y;
    float z;

    float angle = 0f;

    for(int i = 0; i < (segments + 1); i++) {
      angle += 360f / (segments);
      x = Mathf.Sin (Mathf.Deg2Rad * angle) * radius;
      y = 0f;
      z = Mathf.Cos (Mathf.Deg2Rad * angle) * radius;
      trajPath.SetPosition(i,new Vector3(x,y,z));
    }
  }

  void rotateTrajPath() {
    // Angle to rotate from section "Determining the angle to rotate": https://en.wikipedia.org/wiki/Rotation_matrix
    anglesToRotate = new Vector3(satellite.anglesToRotate.x % 360, satellite.anglesToRotate.y % 360, satellite.anglesToRotate.z % 360);

    Quaternion rotX = Quaternion.AngleAxis(anglesToRotate.x, new Vector3(1f,0f,0f));
    Quaternion rotY = Quaternion.AngleAxis(anglesToRotate.y, new Vector3(0f,1f,0f));
    Quaternion rotZ = Quaternion.AngleAxis(anglesToRotate.z, new Vector3(0f,0f,1f));
    this.transform.rotation = rotX * rotY * rotZ;
  }
}
```

7) Save the file and go back to Unity to update the project

8) While OrbitalPath is still selected, select Satellite for Satellite in the Inspector Window



Inspector

OrbitalPath Static

Tag Untagged Layer Default

Transform

Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

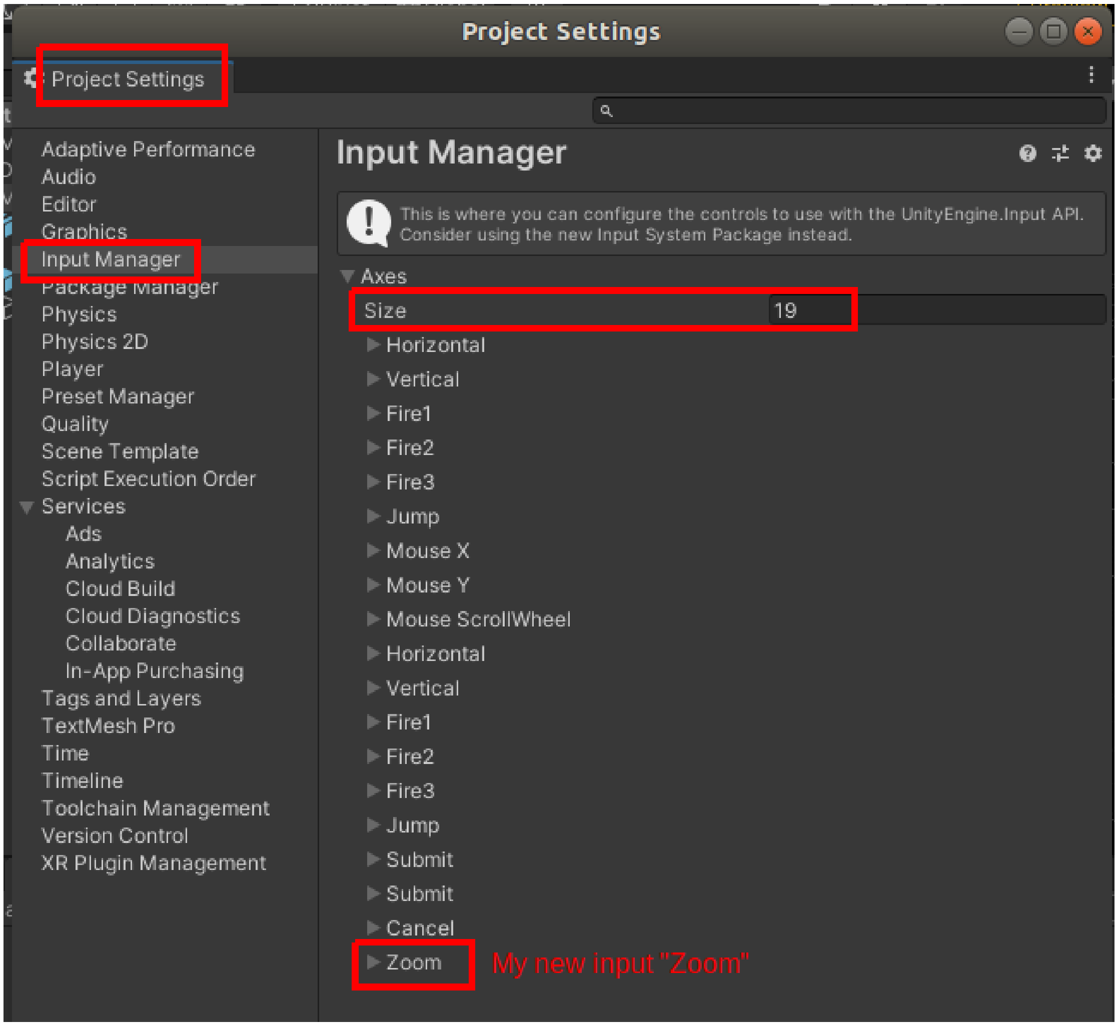
Draw Orbital Path (Script)

Script DrawOrbitalPath
Satellite Satellite (Trajectc
Rotation Axis
X 0 Y 0 Z 0
Angles To Rotate
X 0 Y 0 Z 0

Line Renderer

V. CREATE SCRIPT TO MOVE CAMERA DURING SIMULATIOM

1) Select Main Camera in the Hierarchy
2) Click on Add Component in the Inspector window to add a new script
3) Name the script CameraController
4) Place CameraController.cs in Assets into the Scripts folder
6) Go to **Edit** > **Project Settings** > **Input Manager** and change **Size** to 19 (to add a new input)
7) Select the very last input and rename it to Zoom



Input Manager

This is where you can configure the controls to use with the UnityEngine.Input API. Consider using the new Input System Package instead.

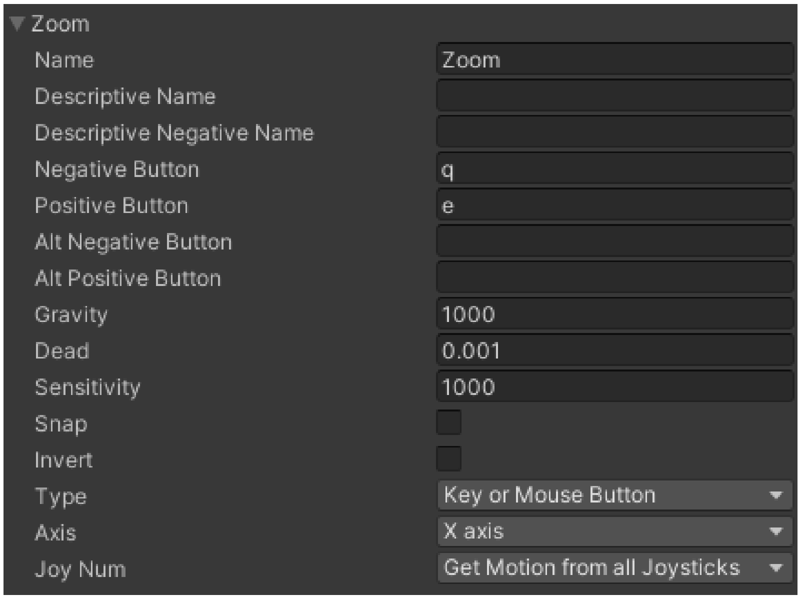
Axes

Size 19

- Horizontal
- Vertical
- Fire1
- Fire2
- Fire3
- Jump
- Mouse X
- Mouse Y
- Mouse ScrollWheel
- Horizontal
- Vertical
- Fire1
- Fire2
- Fire3
- Jump
- Submit
- Submit
- Cancel
- Zoom

My new input "Zoom"

8) Copy the following details for Zoom



Zoom

Name	Zoom
Descriptive Name	
Descriptive Negative Name	
Negative Button	q
Positive Button	e
Alt Negative Button	
Alt Positive Button	
Gravity	1000
Dead	0.001
Sensitivity	1000
Snap	□
Invert	□
Type	Key or Mouse Button
Axis	X axis
Joy Num	Get Motion from all Joysticks

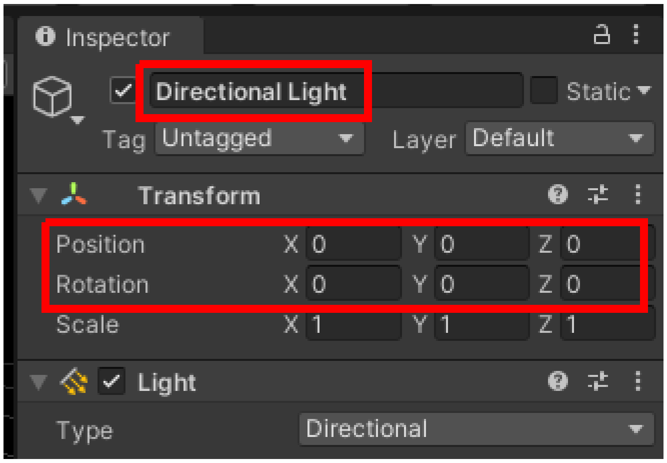
5) Double-click CameraController.cs to open and enter the following code (Note: lines 1-16 is header information)

```
16 using System.Collections;
17 using System.Collections.Generic;
18 using UnityEngine;
19
20 public class CameraController : MonoBehaviour
21 {
22   private float yaw;
23   private float pitch;
24   private float rotSpeed = 20f; // defines rotation speed to rotate camera
25   public float traSpeed = 1000f; // defines translation speed to move camera
26
27   // Start is called before the first frame update
28   void Start()
29   {
30     yaw = transform.eulerAngles.y;
31     pitch = transform.eulerAngles.x;
32   }
33
34   // Update is called once per frame
35   void Update()
36   {
37     // Moves the camera up/down/left/right
38     // To modify keys that are assigned to "Horizontal", "Vertical", "Zoom"
39     // go to Edit > Project Settings > Input Manager
40     // Input Manager ref: https://docs.unity3d.com/Manual/class-InputManager.html
41     // Note: CV added "Zoom" with keys q and e
42     var throwX = Input.GetAxisRaw("Horizontal"); // left/right or a/d keys
43     var throwY = Input.GetAxisRaw("Vertical"); // up/down or w/s keys
44     var throwZ = Input.GetAxisRaw("Zoom"); // q/e keys
45     var changeTraSpeed = Input.GetAxisRaw("Mouse ScrollWheel");
46     traSpeed += changeTraSpeed; //
47     transform.position += transform.right * traSpeed * throwX * Time.deltaTime; // moves camera left/right
48     transform.position += transform.up * traSpeed * throwY * Time.deltaTime; // moves camera up/down
49     transform.position += transform.forward * traSpeed * throwZ * Time.deltaTime; // moves cmeara out/in
50
51     // rotates the camera mouse left-click + drag
52     if(Input.GetMouseButton(0)) // checks if mouse left-click is pressed
53     {
54       yaw += rotSpeed * Input.GetAxis("Mouse X") * Time.deltaTime;
55       pitch += rotSpeed * Input.GetAxis("Mouse Y") * Time.deltaTime;
56       transform.eulerAngles = new Vector3(pitch, yaw, 0f);
57     }
58   }
59 }
```

6) Save the file and go back to Unity to update the project

VI. MODIFYING THE LIGHTING

1) To brighten the planet and the environment, create a new empty object called “Lightings” in the Hierarchy window
2) Place Directional Light under Lightings
3) Change the Position and Rotation to (0, 0, 0)



Directional Light

Static

Tag Untagged

Layer Default

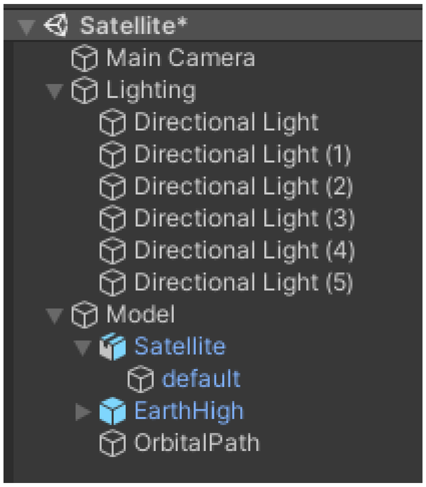
Position X 0 Y 0 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Type Directional

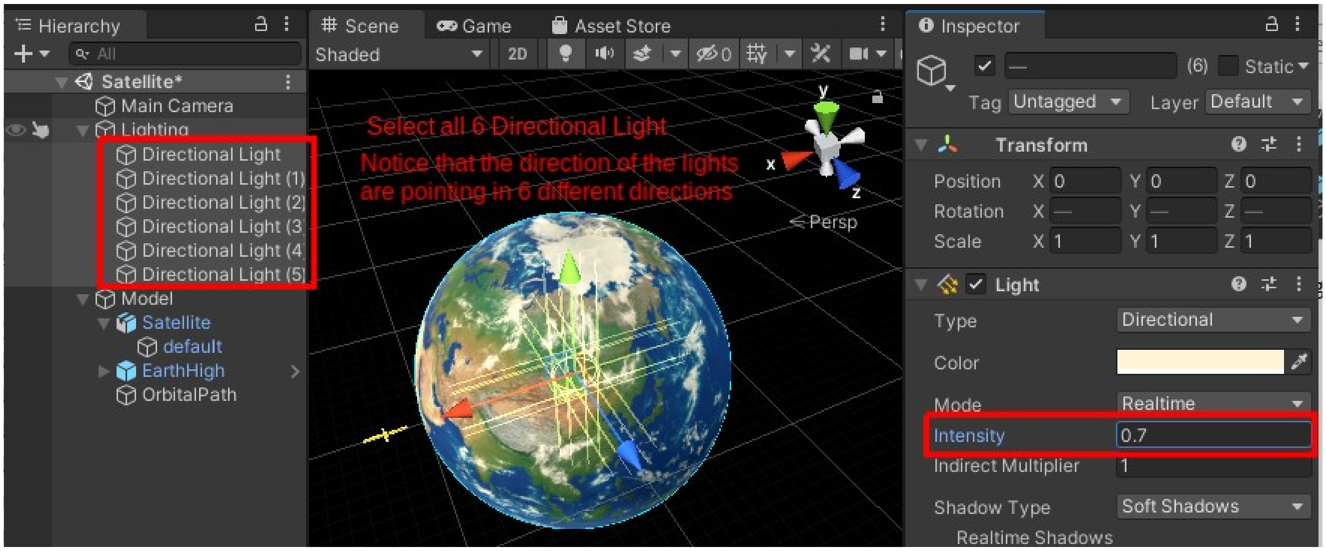
3) Duplicate Directional Light 5 times (total of 6 Direction Lights as shown below) by selecting it and pressing [CTRL]+D



Satellite*
- Main Camera
- Lighting
  - Directional Light
  - Directional Light (1)
  - Directional Light (2)
  - Directional Light (3)
  - Directional Light (4)
  - Directional Light (5)
- Model
  - Satellite
    - default
  - EarthHigh
  - OrbitalPath

4) Change the Rotation for each Direction Light given in the table below

Light Name	Rotation X	Rotation Y	Rotation Z
Directional Light	0	0	0
Directional Light (1)	0	90	0
Directional Light (2)	0	180	0
Directional Light (3)	0	270	0
Directional Light (4)	90	0	0
Directional Light (5)	270	0	0



Select all 6 Directional Light

Notice that the direction of the lights are pointing in 6 different directions

Satellite*
- Main Camera
- Lighting
  - Directional Light
  - Directional Light (1)
  - Directional Light (2)
  - Directional Light (3
  - Directional Light (4
  - Directional Light (5)
- Model
  - Satellite
    - default
  - EarthHigh
  - OrbitalPath

(6) Static
Tag Untagged Layer Default

Transform

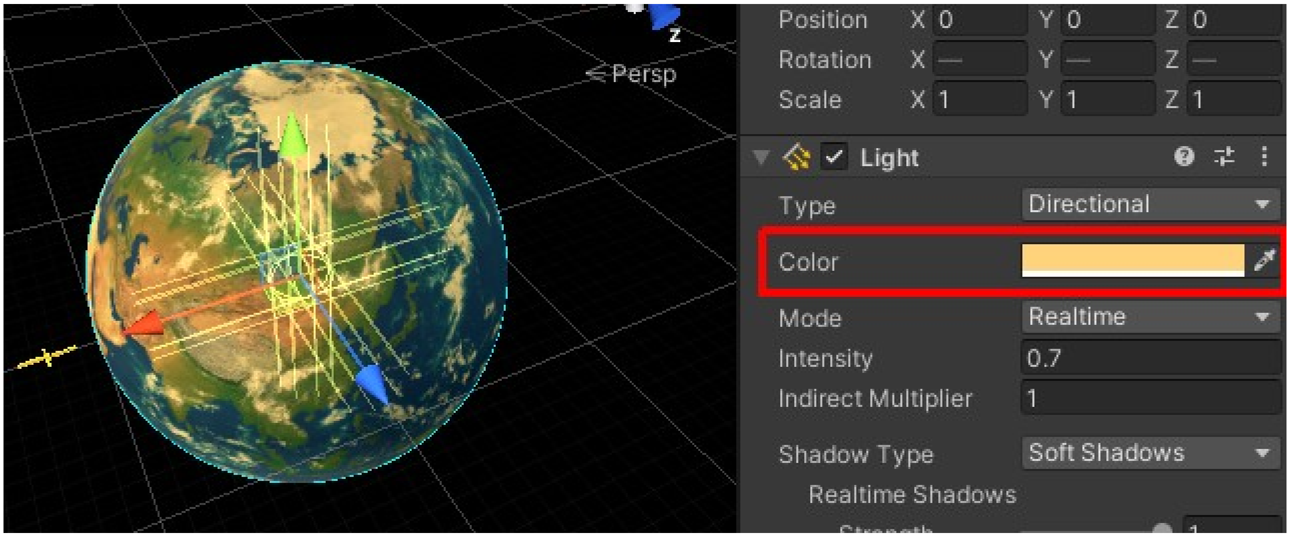
Position	X 0	Y 0	Z 0
Rotation	X —	Y —	Z —
Scale	X 1	Y 1	Z 1

Light

Type	Directional
Color	
Mode	Realtime
Intensity	0.7
Indirect Multiplier	1
Shadow Type	Soft Shadows

Realtime Shadows

5) Select all 6 Direction Lights and change the 0.7 (or a preferred intensity)
6) While all 6 are still selected, change the color to yellow-orange



Position	X 0	Y 0	Z 0
Rotation	X —	Y —	Z —
Scale	X 1	Y 1	Z 1

Light

Type	Directional
Color	
Mode	Realtime
Intensity	0.7
Indirect Multiplier	1
Shadow Type	Soft Shadows

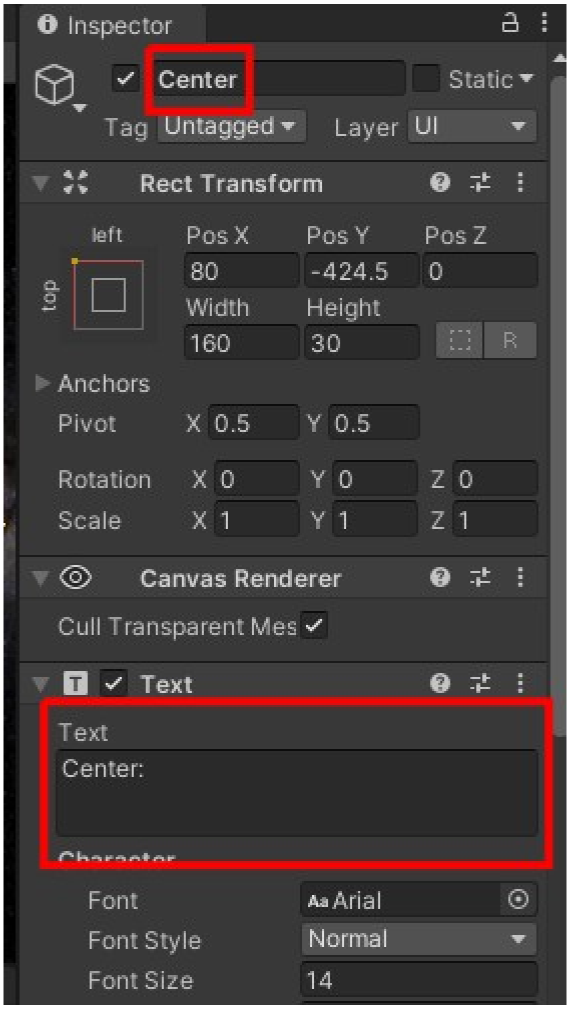
Realtime Shadows

VI. DISPLAY INFORMATION IN GAME VIEW

1) Right-click an empty area in the **Hierarchy** to add **UI** > **Text**
2) Repeat this 7 times and rename each Text object in the following list
Note: For each Text object, enter the following Text in Text

Text Object Name	Text
Center	Center:
Radius	Radius:
Incline	Incline:
Speed	Satellite Rotation Speed:
Rotation Axis	Rotation Axis:

Cross Vector	Cross Vector:
Perpendicular Vector	Perpendicular Vector:


Inspector

☑ Center

Static

Tag Untagged

Layer UI

Rect Transform

left

top

Pos X	Pos Y	Pos Z
80	-424.5	0
Width	Height	
160	30	R

Anchors

Pivot X 0.5 Y 0.5

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Canvas Renderer

Cull Transparent Mes ✓

☑ Text

Text

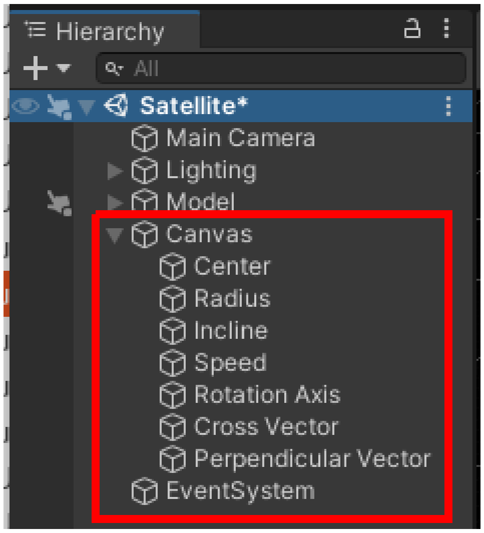
Center:

Character

Font Arial

Font Style Normal

Font Size 14

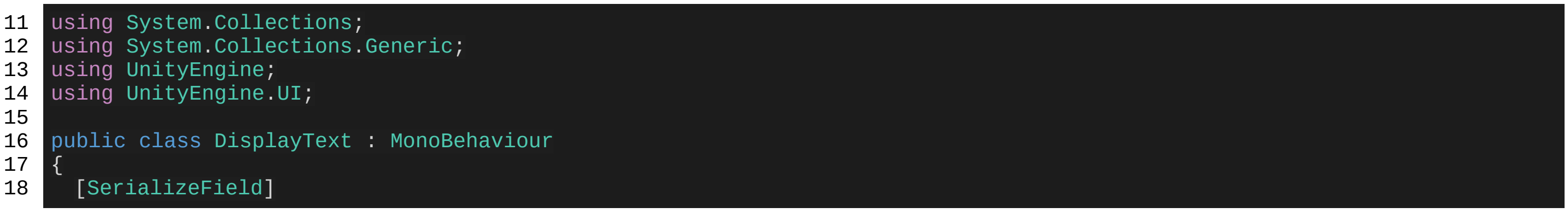

Hierarchy

All

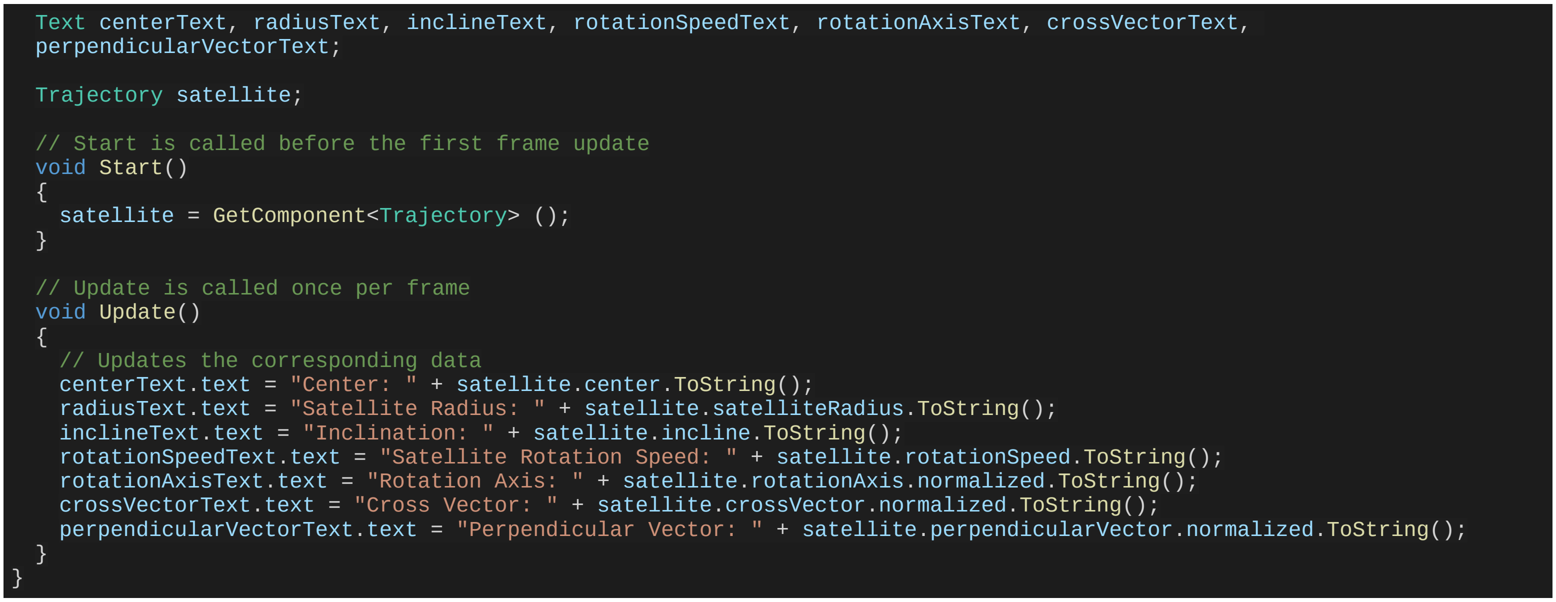
```
Satellite*
    Main Camera
    Lighting
    Model
    Canvas
        Center
        Radius
        Incline
        Speed
        Rotation Axis
        Cross Vector
        Perpendicular Vector
    EventSystem
```

3) Select the **Satellite** object in the **Hierarchy** add a new C# script
4) Name the file **DisplayText** and move the file to the **Scripts** folder
Note: By this time there should be 4 scripts
(1) CameraController.cs
(2) DisplayText.cs
(3) DrawOrbitalPath.cs
(4) Trajectory.cs

5) Enter the following



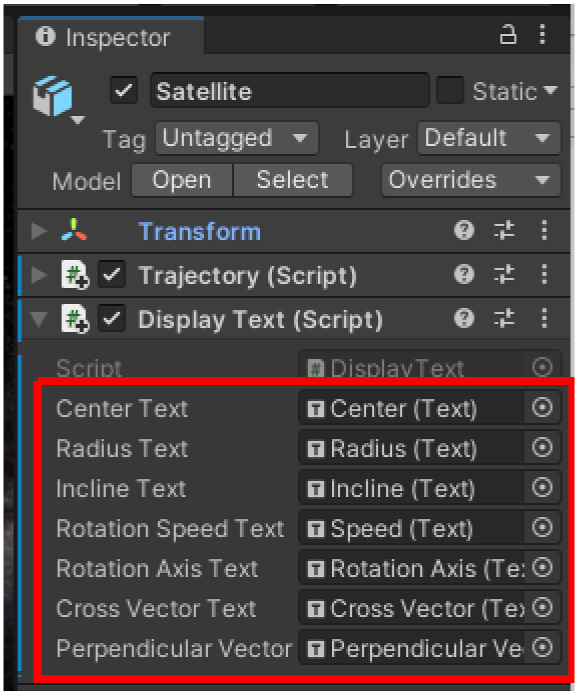
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DisplayText : MonoBehaviour
{
  [SerializeField]
```



```
  Text centerText, radiusText, inclineText, rotationSpeedText, rotationAxisText, crossVectorText,
  perpendicularVectorText;

  Trajectory satellite;

  // Start is called before the first frame update
  void Start()
  {
    satellite = GetComponent<Trajectory> ();
  }

  // Update is called once per frame
  void Update()
  {
    // Updates the corresponding data
    centerText.text = "Center: " + satellite.center.ToString();
    radiusText.text = "Satellite Radius: " + satellite.satelliteRadius.ToString();
    inclineText.text = "Inclination: " + satellite.incline.ToString();
    rotationSpeedText.text = "Satellite Rotation Speed: " + satellite.rotationSpeed.ToString();
    rotationAxisText.text = "Rotation Axis: " + satellite.rotationAxis.normalized.ToString();
    crossVectorText.text = "Cross Vector: " + satellite.crossVector.normalized.ToString();
    perpendicularVectorText.text = "Perpendicular Vector: " + satellite.perpendicularVector.normalized.ToString();
  }
}
```

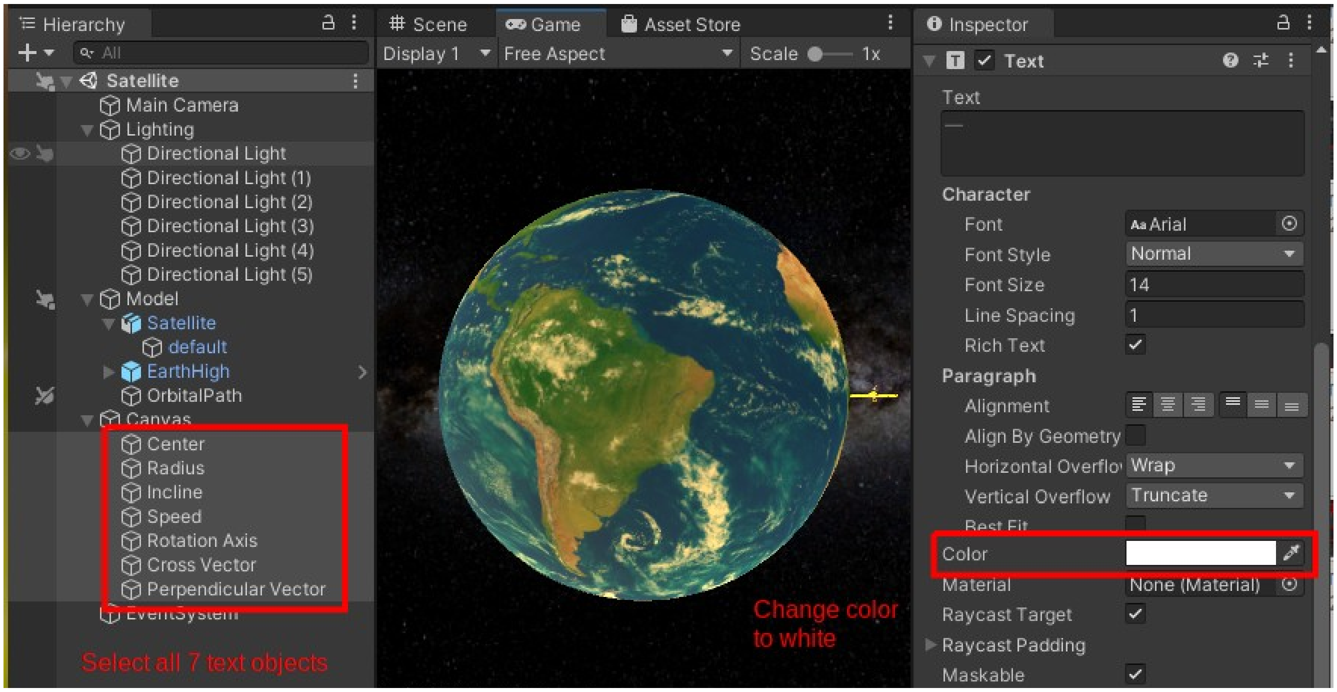
6) Save the file and return to Unity to update the project.

7) Select the **Satellite** object, and for each of the items in the **Display Text** component select the object as shown below



Script: DisplayText
Center Text: Center (Text)
Radius Text: Radius (Text)
Incline Text: Incline (Text)
Rotation Speed Text: Speed (Text)
Rotation Axis Text: Rotation Axis (Te:
Cross Vector Text: Cross Vector (Te)
Perpendicular Vector: Perpendicular Ve

8) Select all 7 text objects and change the **Color** to white



Text

Character

Font: Arial

Font Style: Normal

Font Size: 14

Line Spacing: 1

Rich Text: ✓

Paragraph

Alignment

Align By Geometry: □

Horizontal Overflow: Wrap

Vertical Overflow: Truncate

Best Fit: □

Color

Material: None (Material)

Raycast Target: ✓

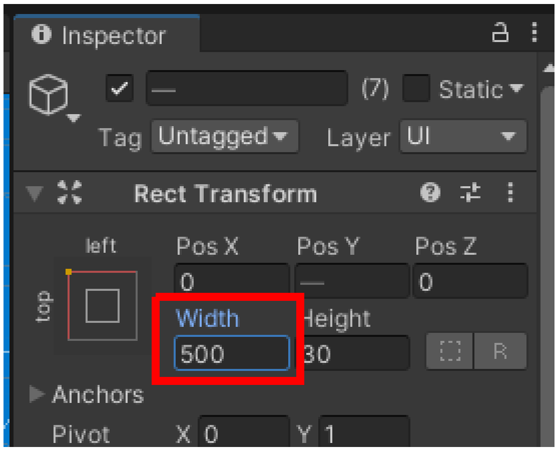
Raycast Padding

Maskable: ✓

Select all 7 text objects

Change color to white

9) While all the text objects still selected change the Width to 500 (this guarantees that none of the text will be truncated)



☑ — (7) □ Static

Tag: Untagged

Layer: UI

Pos X: 0

Pos Y: —

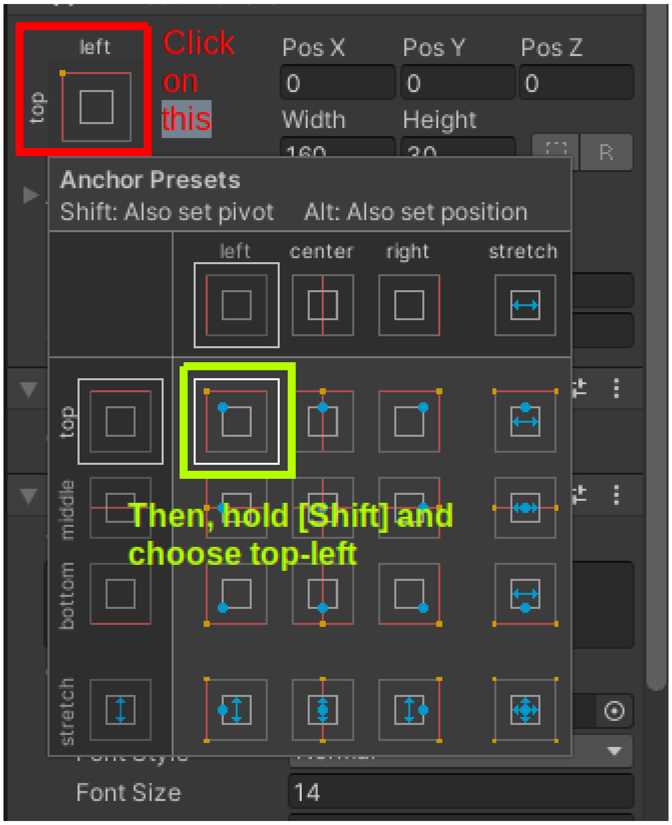
Pos Z: 0

Width: 500

Height: 30

Pivot: X 0 Y 1

10) While holding [Shift], click on the two squares shown below and choose the top-left (A blue dot will appear if [Shift] is pressed) to change the position of the text to top-left corner



Click on this

Then, hold [Shift] and choose top-left

Anchor Presets

Shift: Also set pivot Alt: Also set position

Pos X 0

Pos Y 0

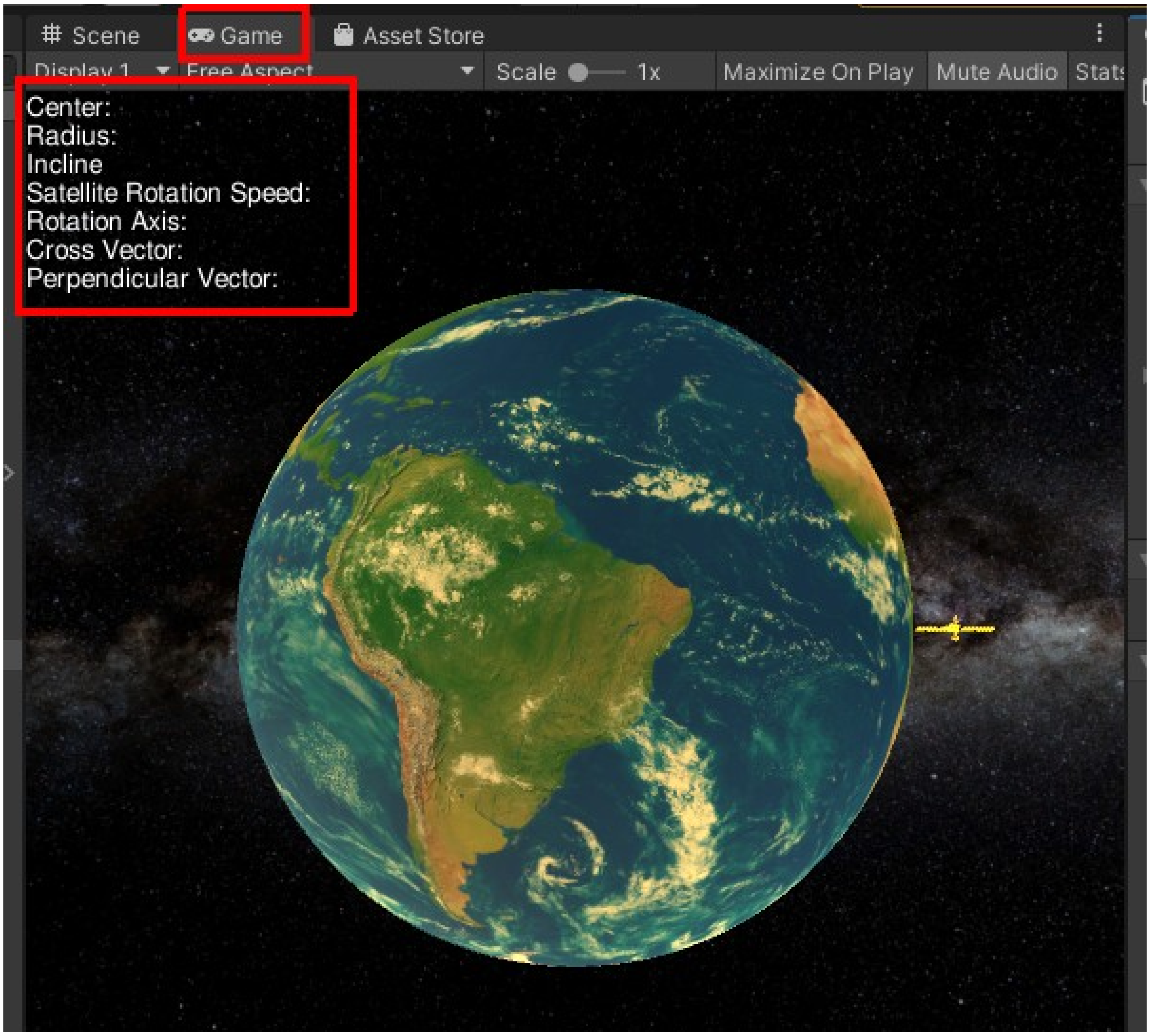
Pos Z 0

Font Size 14

11) For each change the position POS X/Y/Z to the following table such that they are not overlapping

Text Object	POS X	POS Y	POSZ
Center	0	0	0
Radius	0	-15	0
Incline	0	-30	0
Speed	0	-45	0
Rotation Axis	0	-60	0
Cross Vector	0	-75	0
Perpendicular Vector	0	-90	0

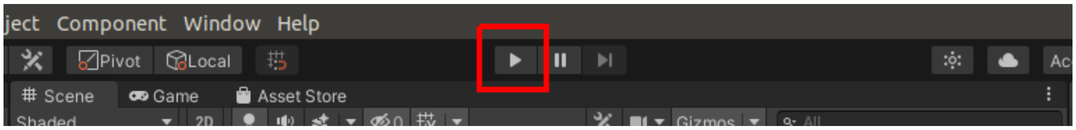
12) Go to Game view to see the text located in the top-left corner



Center:
Radius:
Incline
Satellite Rotation Speed:
Rotation Axis:
Cross Vector:
Perpendicular Vector:

VII. PLAY THE SIMULATION

Click on the play button to see the simulation.



1) You can change the Satellite **Radius, Rotation Speed, Incline** to any value in the Satellites' Inspector window.

2) Pressing the follow keys will move the camera

Keys	Movement
A	Left
D	Right
W	Up

S	Down
Q	Zoom Out
E	Zoom In
Mouse Left-Click Drag	Rotate camera in direction of mouse movement

(END)