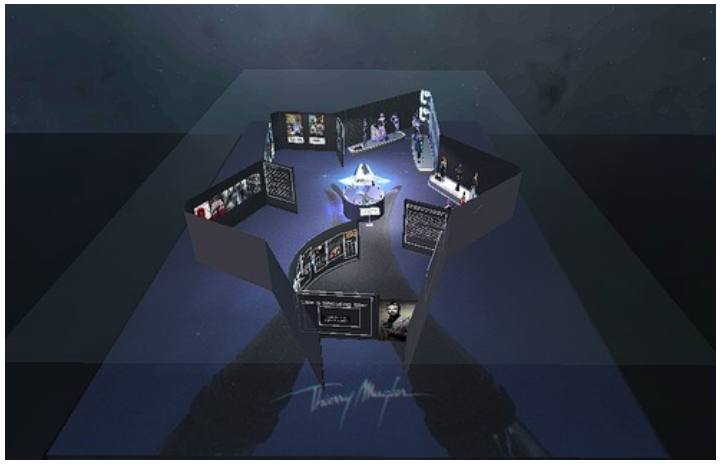
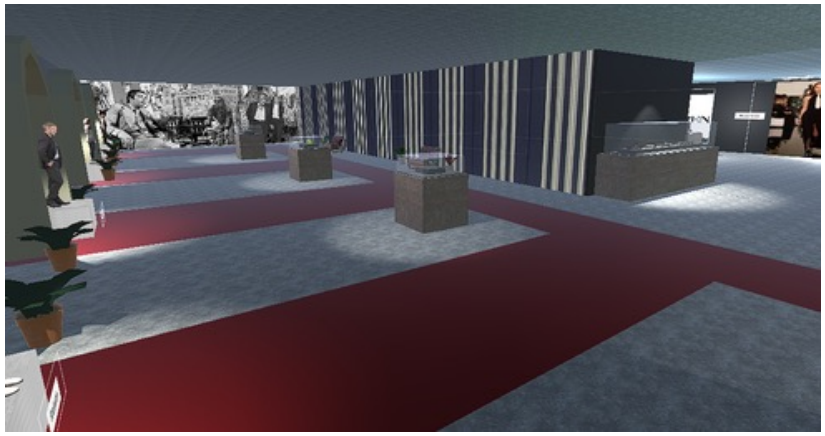
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

How to create a virtual Environment with Unity

Shirin Hajahmadi

Department of Computer Science and Engineering





These are ***Unity example projects*** in VARLAB website from previous students:

https://site.unibo.it/varlab/en/students/projects-2

Links to the resources:

GitHub Unity template project link:

https://github.com/shirinhaj/WorkshopExample

Google Drive Link:

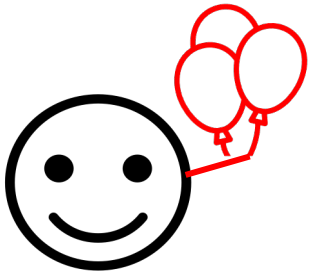
https://drive.google.com/drive/folders/1ghdS9nVc0EgZLrZdWDrgxH5SDD8B48IN?usp=sharing

Plan for today:

3rd of July 2024.

Unity session

10:00 - 12:00	Unity for creating virtual environments (essentials) Shirin Hajahmadi, Giacomo Vallasciani
12:00 - 13:00	Lunch
13:00 - 17:00	Implementation of an example Unity project. Shirin Hajahmadi, Giacomo Vallasciani



Part 1 : Unity for creating virtual environments (essentials)

Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- Working with Lights
- Setting the Skybox

Part 1 : Unity for creating virtual environments (essentials)

- **Introduction to Unity as a Game Engine**
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- Working with Lights
- Setting the Skybox

Unity

- A famous real-time development Game Engine [1].
- You can create (2D / 3D) applications.
- It supports building the application on different platforms like smartphones, desktop, Virtual Reality headsets, and Mixed reality headsets.





1. A game engine is a software framework that provides a suite of tools and features that help developers create games more efficiently.

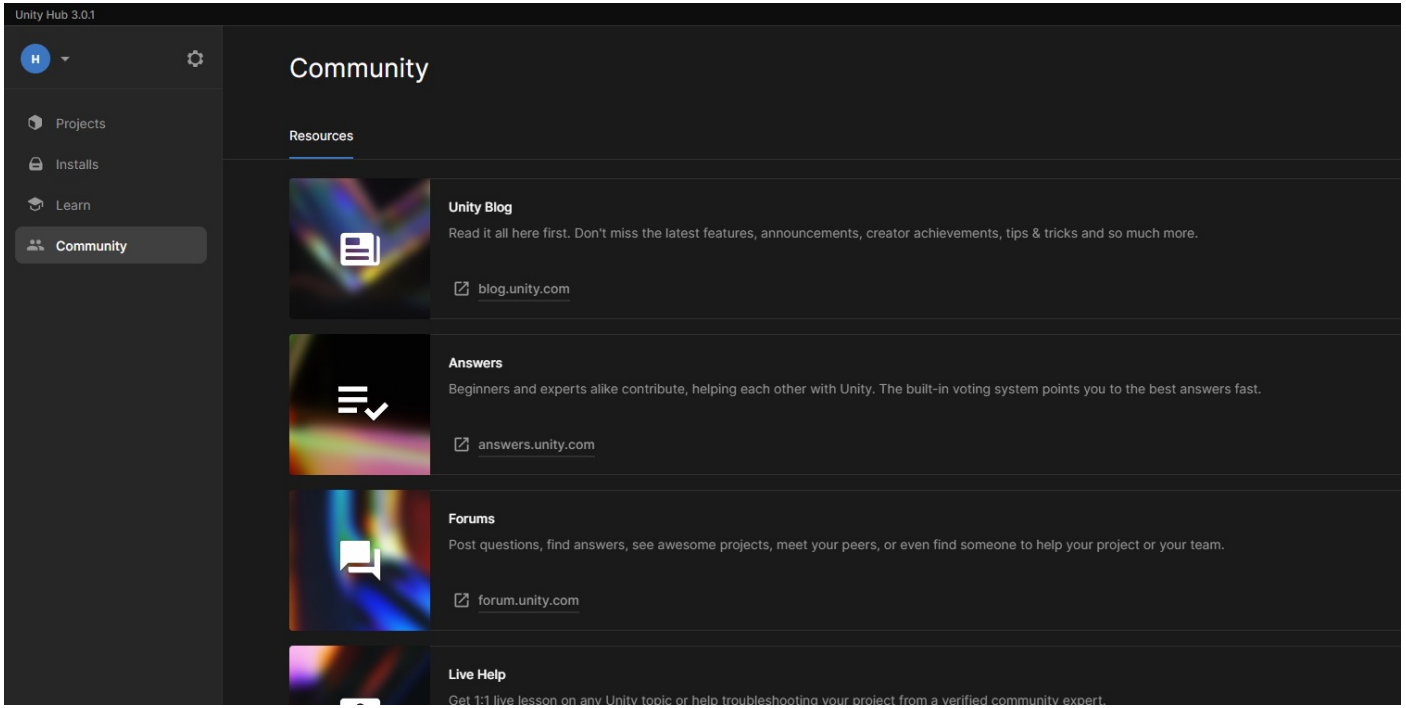
Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- **Getting Started with Unity**
  - **Introduction to Unity Hub**
  - **Download and Install Unity Hub**
  - **Create an Account and Sign In**
  - **Activate the License**
  - **Install the Unity Editor**
  - **Create a New Unity Project with Unity Hub**
  - **Work with Unity Interfaces**
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- Working with Lights
- Setting the Skybox

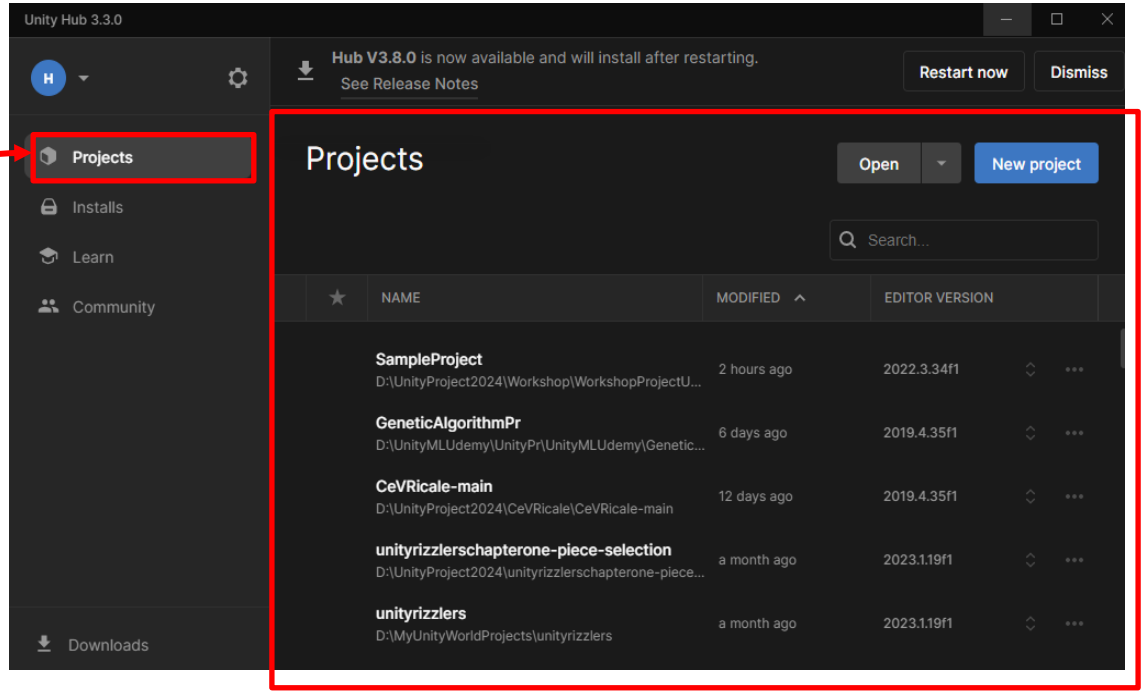
Introduction to Unity Hub:

We use the **Unity Hub** to manage:

- Creating new projects.
- Accessing to our previous projects.
- Multiple installations of the **Unity Editor.**
- Learning Unity.
- Collaboration, support, discussions, issues, sponsorships.
- Activating the license.



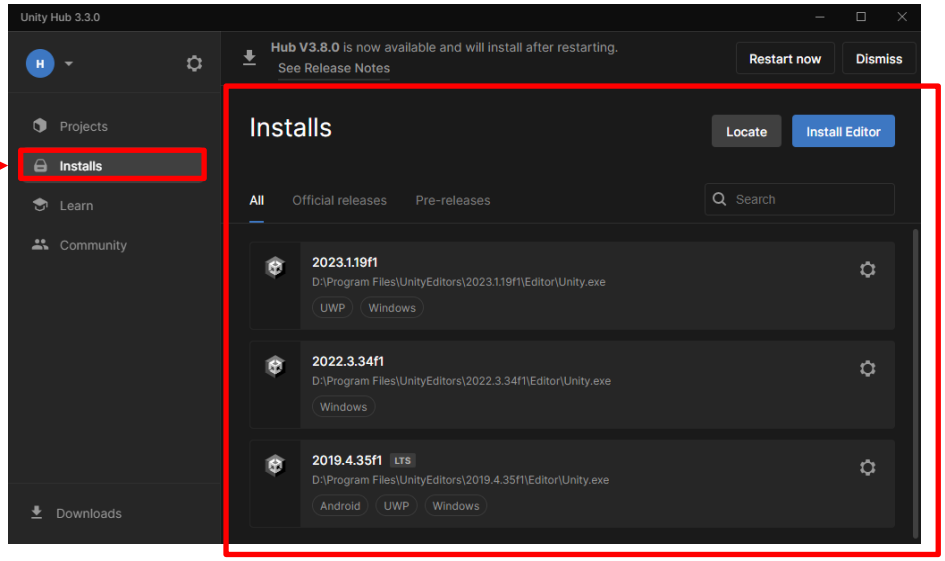
Introduction to Unity Hub:



The **Projects tab** is where you can create, import, and manage new and existing projects.

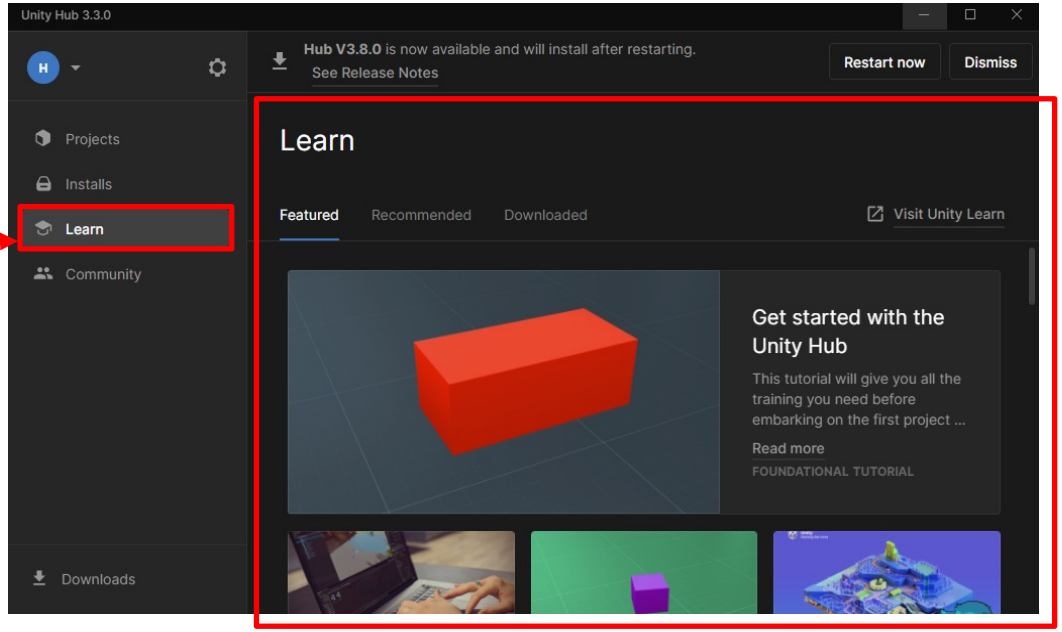
Introduction to Unity Hub:

The **Installs tab** allows you to manage and configure various Unity editor installations in the Hub.



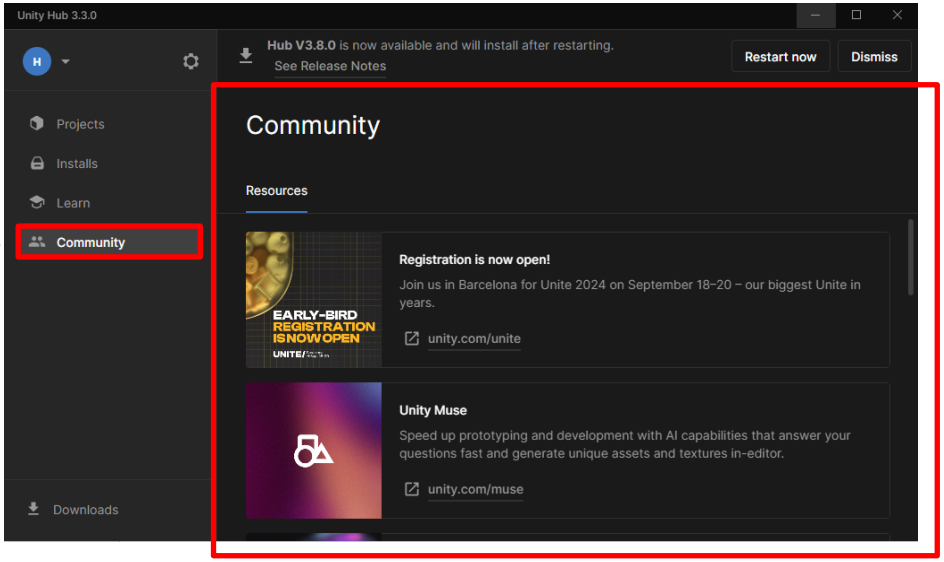
Introduction to Unity Hub:

The **Learn** tab offers instant access to featured learning experiences and tutorials to help you master specific Unity skills.



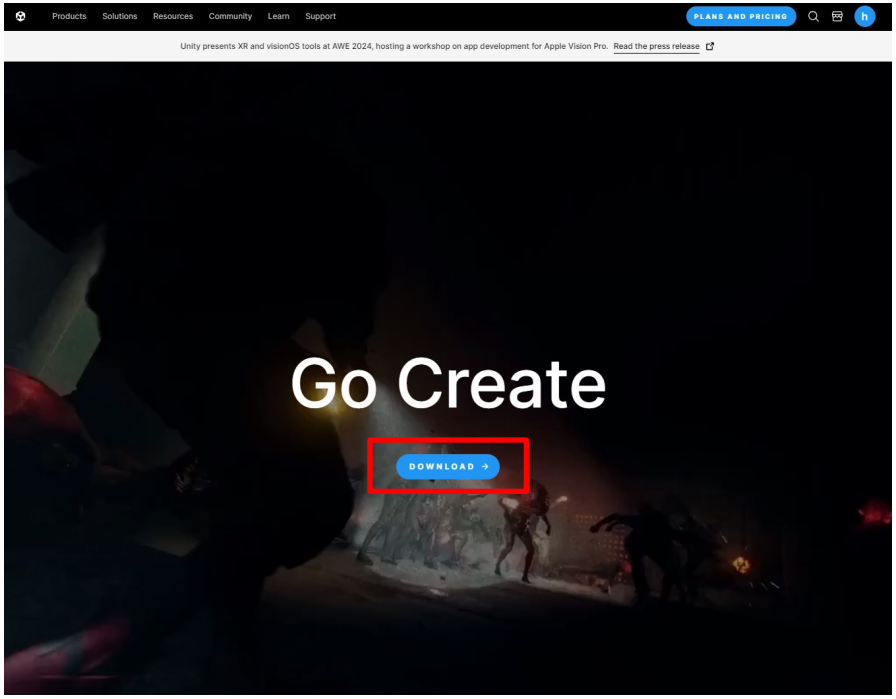
Introduction to Unity Hub:

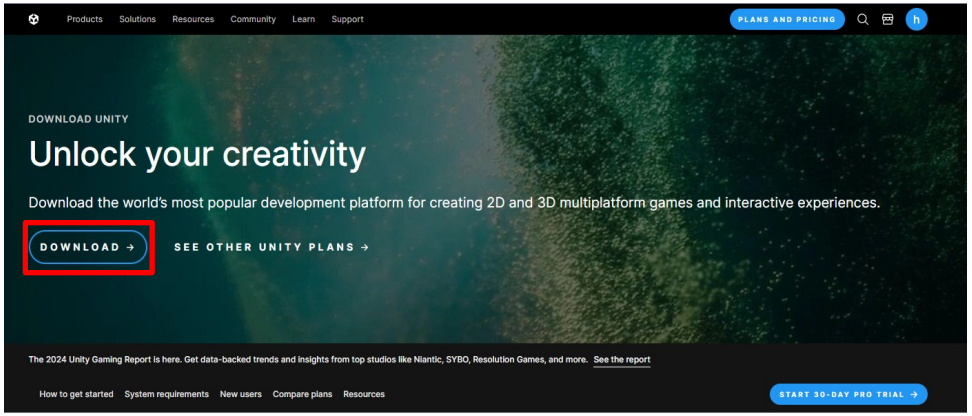
The **Community tab** provides a quick reference to various platforms where you can engage and connect with the broader Unity community.



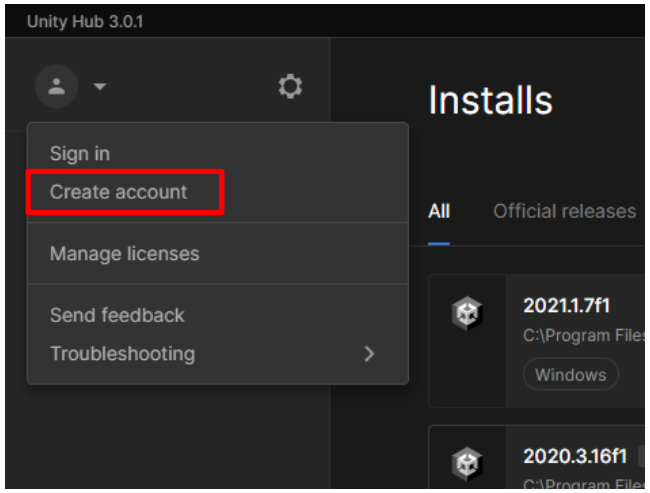
Download and Install Unity hub:

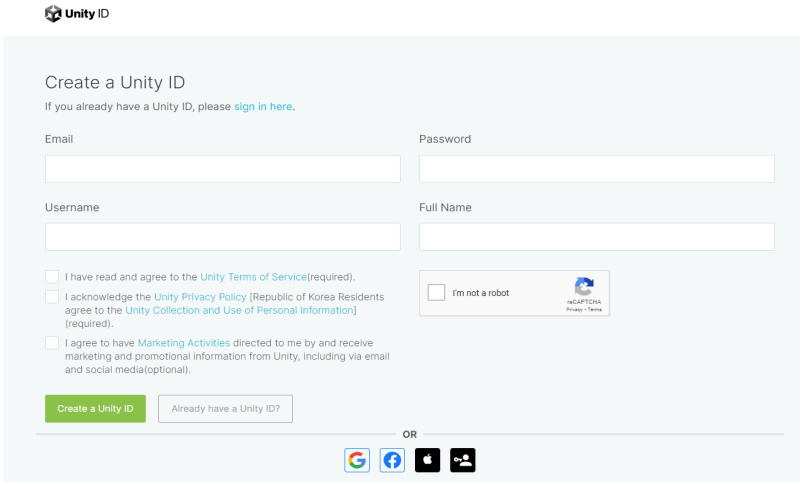
From this link: https://unity.com

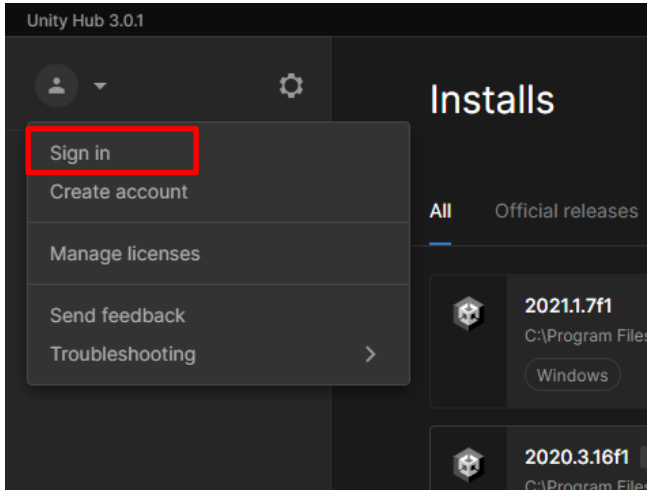


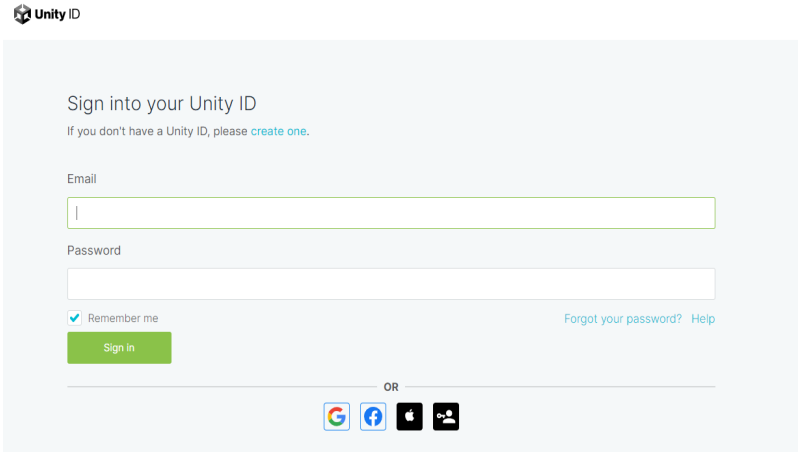


Create an account and sign in into the Unity Hub:





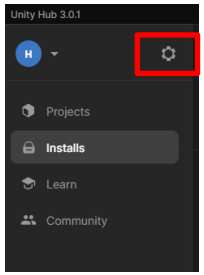


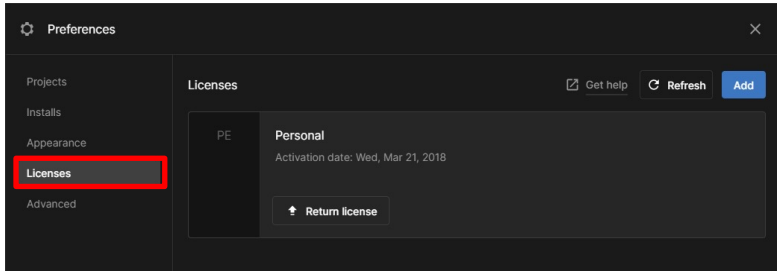


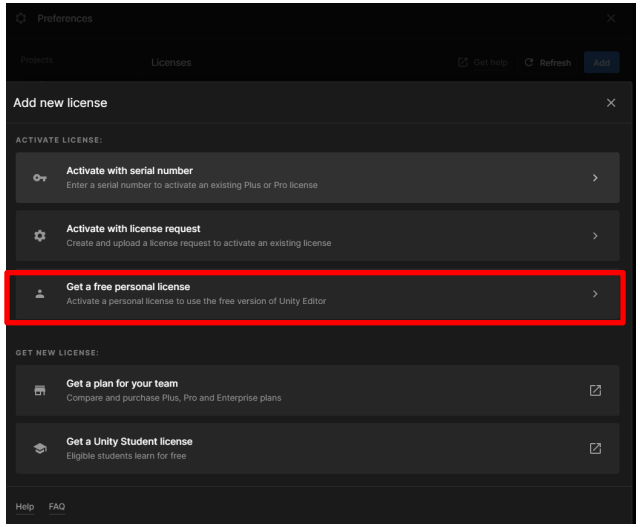
Activate the License:

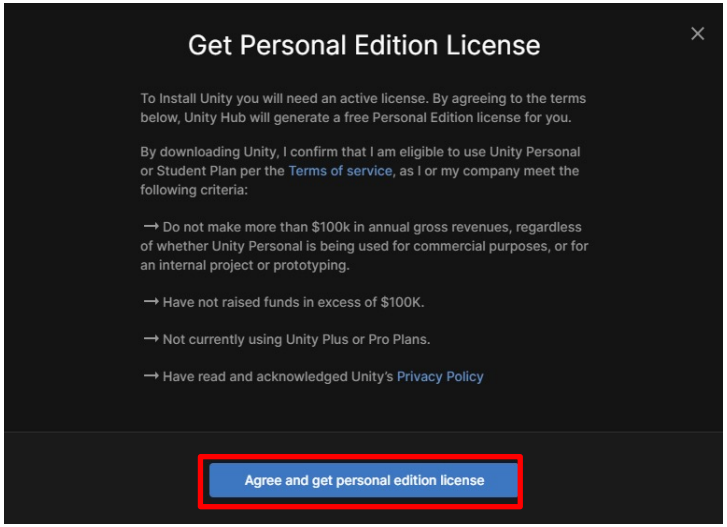
1. Click on the gear on the left of the **Unity Hub**.
2. Choose the **Licenses** tab.
3. Click on the **Add** button.
4. Click on the **Get a free personal license**.
5. Click on the **Agree and get personal edition license**.

You can also activate the License manually by following this Link:
https://support.unity.com/hc/en-us/articles/4401914348436-How-can-I-manually-activate-my-license-inside-the-hub-





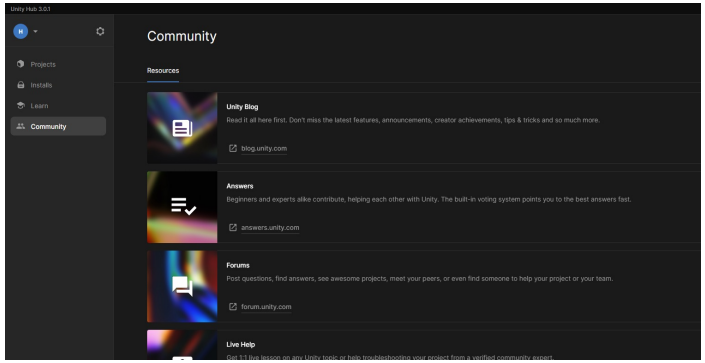


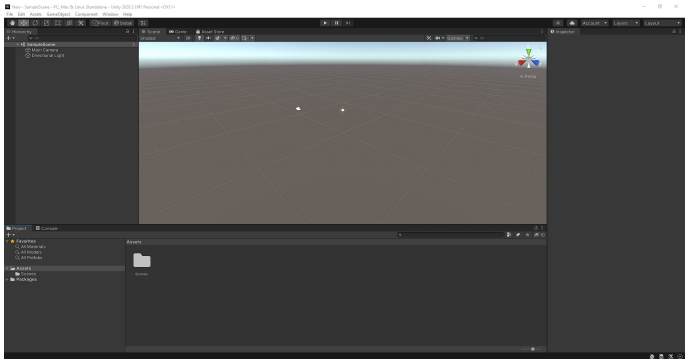


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

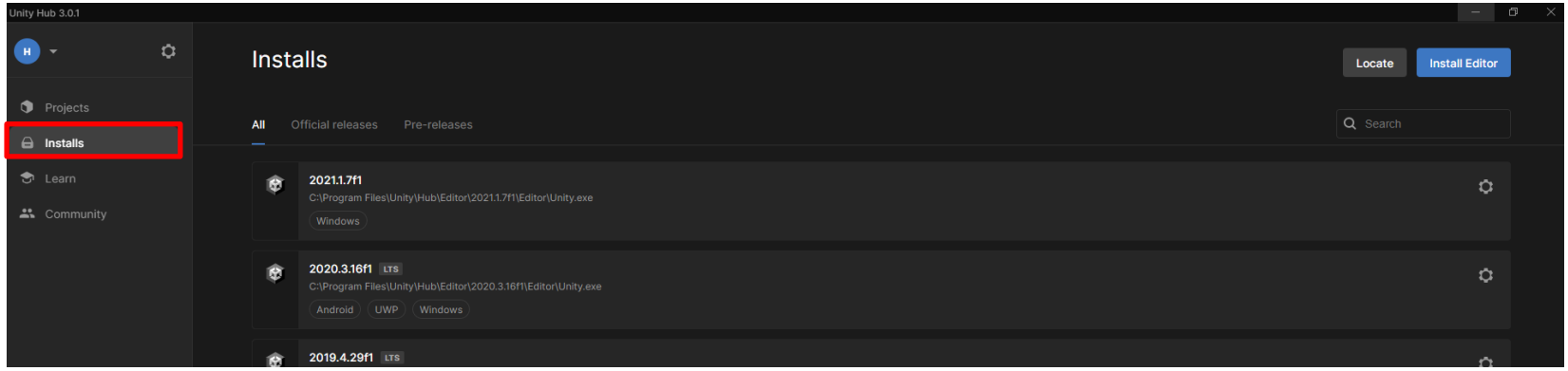
Install a Unity Editor:

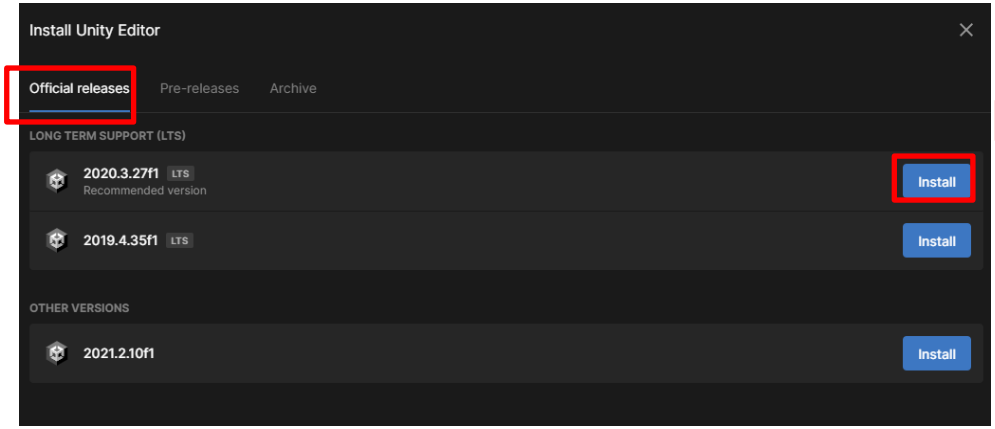
To create virtual experiences, we use the **"Unity Editor"**, which can be installed via the **"Unity Hub"**.

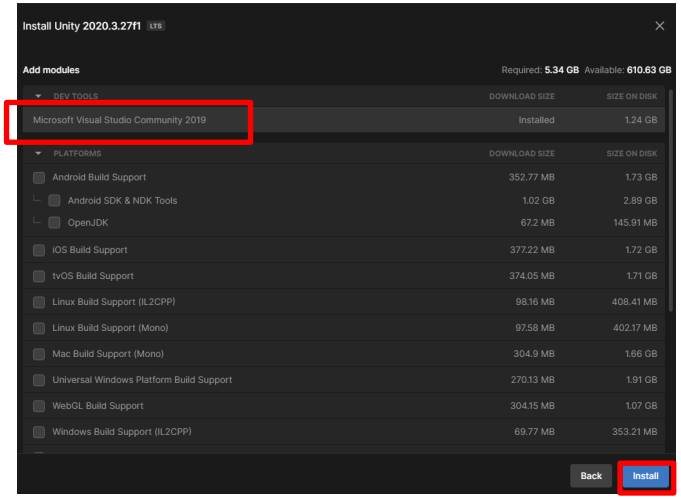




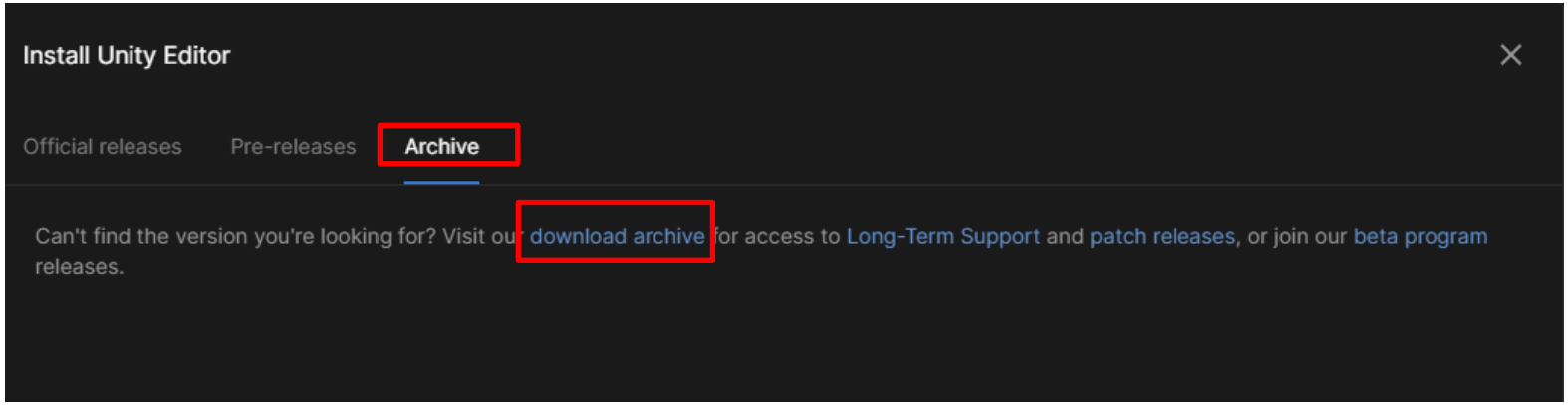
Install a Unity Editor (with suggested options from unity hub):

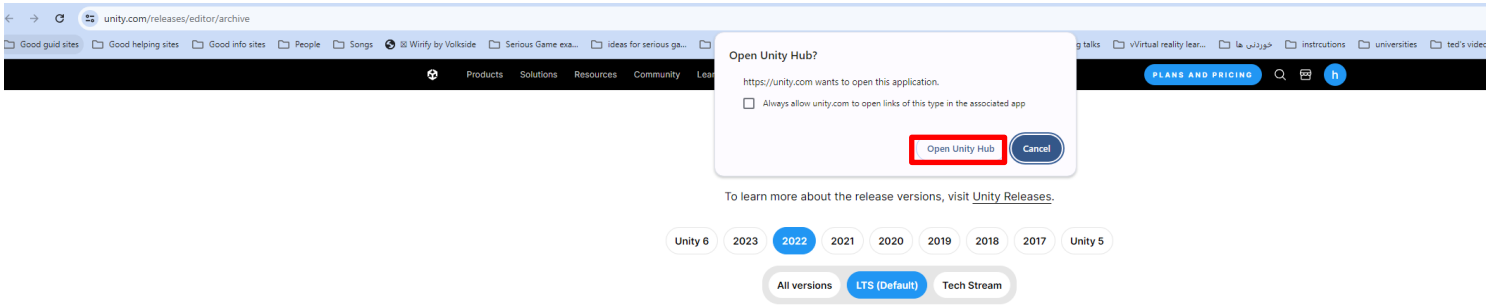




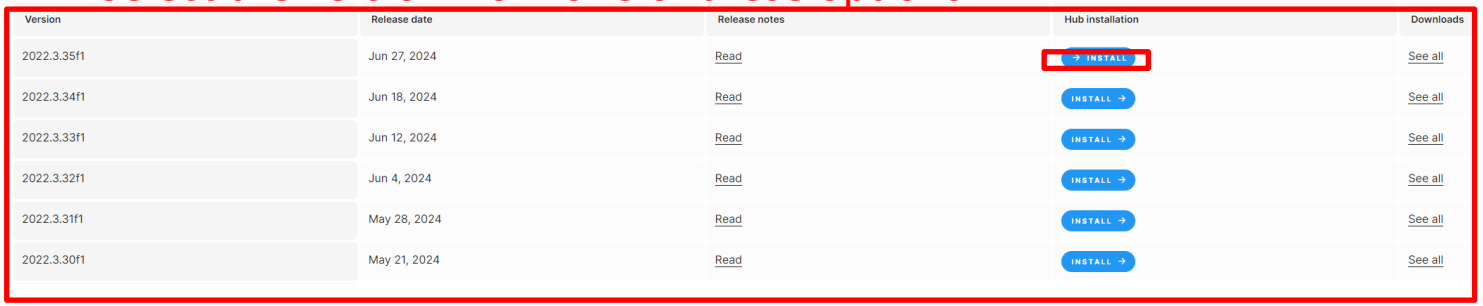


Install a Unity Editor (from archive):

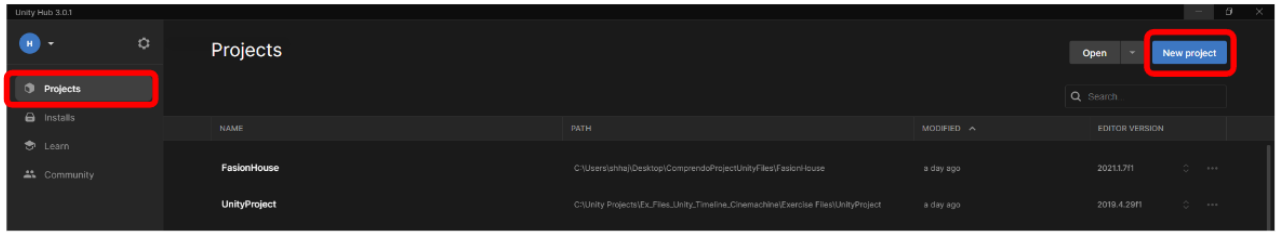




Select the version from one of these options



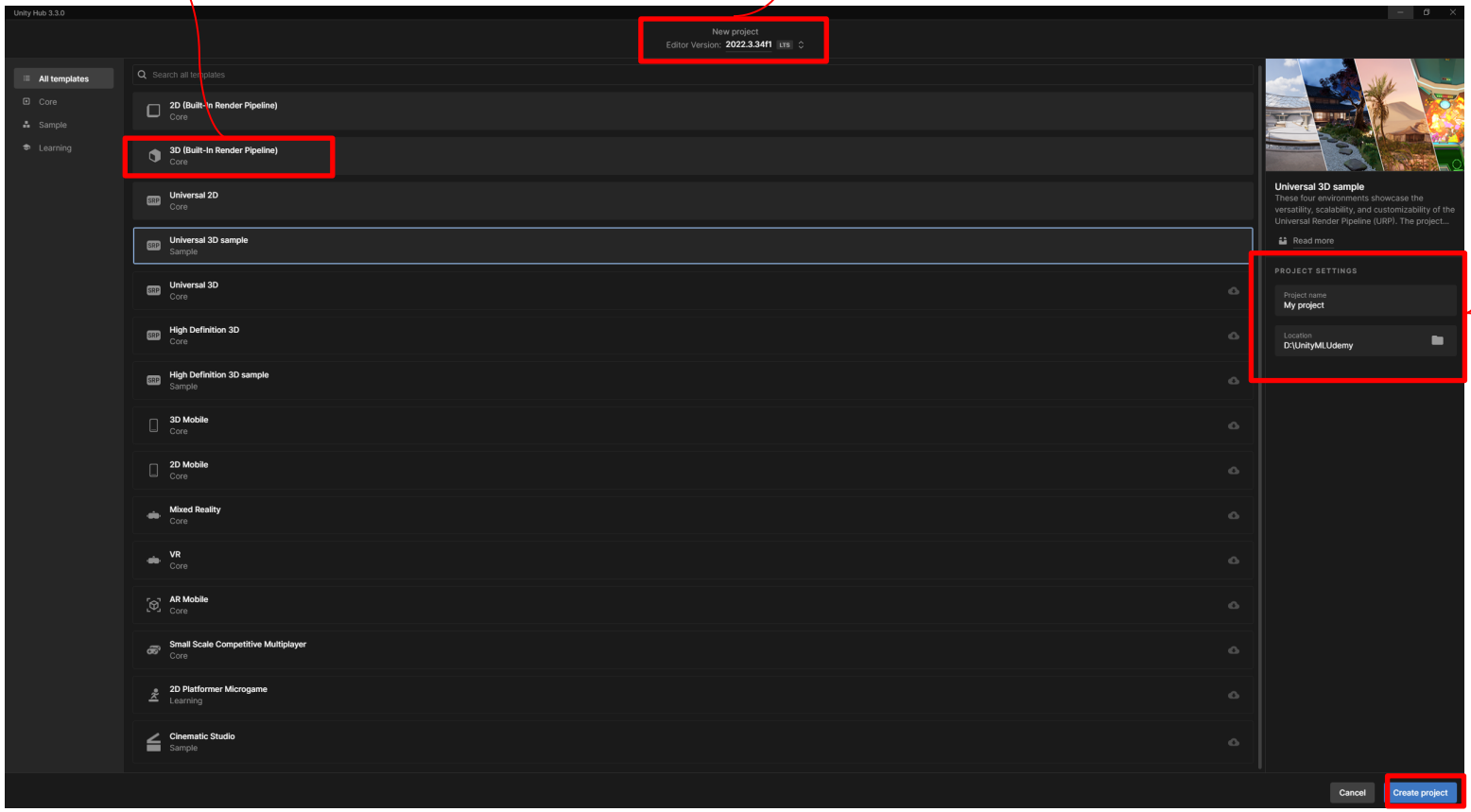
Create a new project with Unity Hub:



Select the Template type

Select the Version of Unity

Unity: 2022.3.35f1



Select the name

Select the location

Work with Unity Interfaces:

Work with Unity Interfaces:

- **The Scene view** is your interactive view into the world you are creating. You can use the **Scene view** to select and position scenery, characters, Cameras, lights, and all other types of Game objects.

- **The Project window** displays all of the files related to your project and is the main way you can navigate and find Assets and other project files in your application.

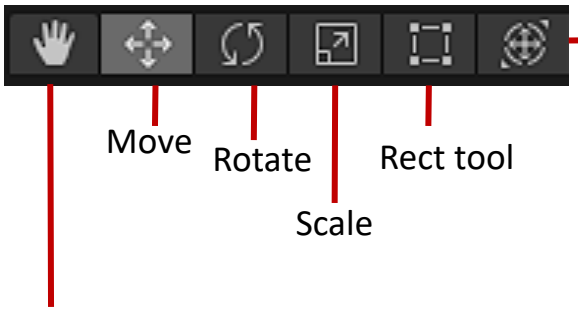
- **The Game view** is rendered from the Camera(s) in your application. It represents your final, published application. You need to use one or more Cameras to control what the player sees when they are using your application.

- **The Inspector window** could be used to view and edit properties and settings for almost everything in the Unity Editor, including Game objects, Unity components, Assets, and Materials.

- **Console Window** displays errors, warnings, and other messages the Editor generates. These errors and warnings help you find issues in your project, such as script compilation errors. They also alert you to actions the Editor has taken automatically, such as replacing missing meta files, which could cause an issue somewhere else in your project.

- **The Hierarchy window** displays every Game object in a Scene, such as Models, Cameras, or Prefabs. You can use the Hierarchy window to sort and group the Game objects you use in a Scene. When you add or remove Game objects in the Scene view, you also add or remove them from the Hierarchy window. The Hierarchy window can also contain other Scenes, with each scene containing their own Game objects.

Work with Unity Interfaces:



Moving the editor's view of the scene without affecting the actual game camera or the objects' positions within the game.

Control:	Description:
Move	Click and drag to move the Camera around.
Orbit	Hold **Alt** (Windows) or **Option** (macOS), and left-click and drag to orbit the Camera around the current pivot point. This option is not available in 2D mode, because the view is orthographic.
Zoom	Hold **Alt** (Windows) or **Option** (macOS), and right-click and drag to zoom the Scene view. On macOS, you can also hold **Control**, and left-click and drag instead.

Part 1 : Unity for creating virtual environments (essentials)

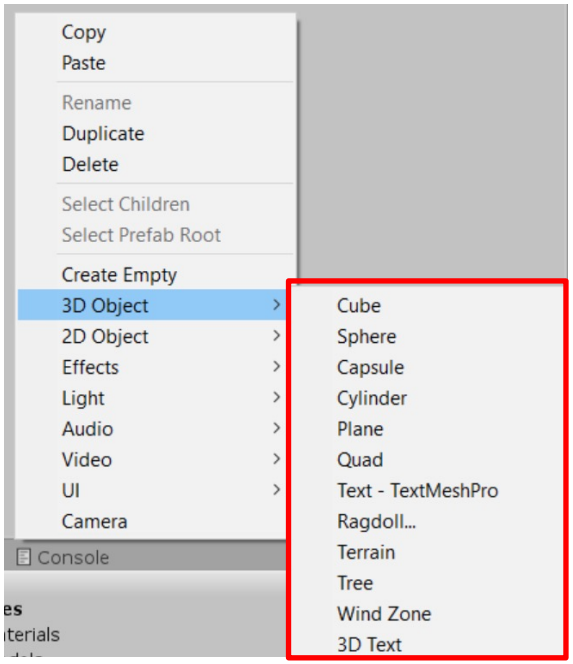
- Introduction to Unity as a Game Engine
- Getting Started with Unity
- **Creating and Managing Basic Game Elements**
  - Create simple Game Objects
  - Work with Supporting Assets
  - Work with Package Manager
  - Work with material
  - Work with components
  - Introducing the Transform and Collider components.
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- Working with Lights
- Setting the Skybox

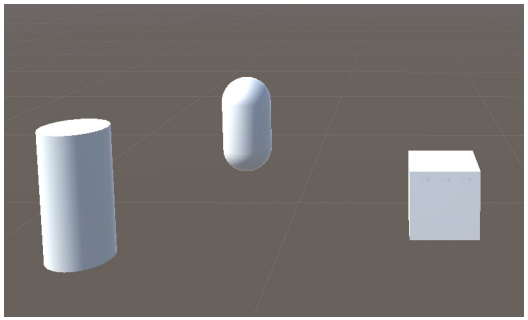
Create simple game objects (for example 3D Objects)

There are a number of primitive object types that can be created directly within Unity, namely the Cube, Sphere, Capsule, Cylinder, Quad and Plane. By transforming and assembling primitive forms, we can easily build several objects.

How to create a Game Object:

1. Right click on the **Hierarchy** window.
2. Select **3D Object**.
3. You can find sphere, cube, capsule, cylinder, etc. to select and create the new Game Object.





Work with Supporting Assets

- Unity can work with 3D models of any shape that can be created with modelling software (3D Max, Blender, Sketch up etc.,).
- Unity can read the following standard 3D file formats (.Fbx, .dae, .dxf, .obj).
- You can check this link for finding the formats that are compatible with Unity.
  https://docs.unity3d.com/Manual/3D-formats.html

Some of the websites for finding the free Assets:

For 3D models:

- Sketchfab: https://sketchfab.com/feed
- Poly Haven: https://polyhaven.com/
- Asset Store of Unity https://assetstore.unity.com/
- Turbosquid https://www.turbosquid.com/
- Free3D https://free3d.com/

For audio:

Free sound:
https://freesound.org/browse/

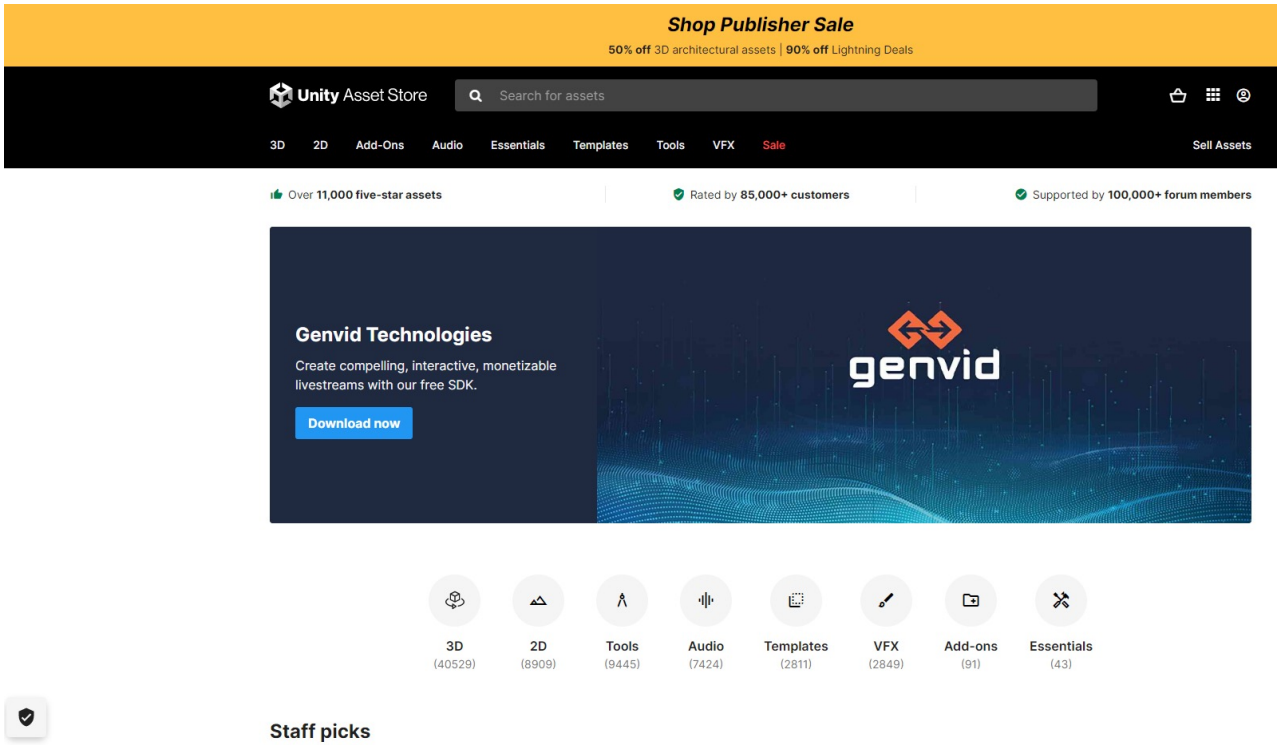
Pixabay:
https://pixabay.com/sound-effects/search/rain/

BBC Sound Effect:
https://sound-effects.bbcrewind.co.uk/search

Unity Asset Store

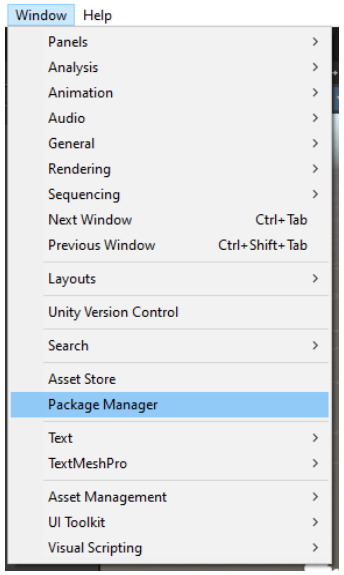
- The Unity **Asset Store** contains a library of free and commercial assets that Unity Technologies as well as members of the community create. A wide variety of assets are available, including Textures, Models, animations, entire project examples, tutorials, and Editor extensions.

- Starting from Unity 2020.1, the dedicated Asset Store window is no longer hosted inside the Unity Editor. However, you can still access the Asset Store website at https://assetstore.unity.com/ and you can still search for your purchased and downloaded Asset store packages, as well as import and download them directly in the **Package Manager** window.



Package Manager

- The Unity Package Manager is a tool that helps you manage project dependencies and extensions. It allows you to add, update, and remove packages to and from your Unity project, providing access to features, assets, and tools created by Unity and the community.



- The Unity Package Manager organizes packages into several sections to help you manage project dependencies effectively: **Unity Registry, My Assets, In Project, Built-In.**

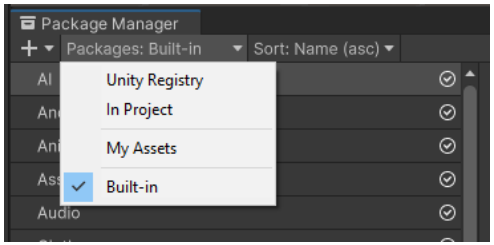
  - **Unity Registry**
    - The Unity Registry contains official packages provided by Unity Technologies.
    - **Examples**: Cinemachine, ProBuilder, Post Processing.
  - **My Assets**
    - The My Assets section includes packages and assets you have purchased or downloaded from the Unity Asset Store. Some assets listed in this section may not be installed in your current project yet but are available for download and installation.
    - **Examples**: Asset Store plugins, 3D models, sound effects.
  - **In Project:**
    - The In Project section lists all packages currently installed in your Unity project.
    - **Examples**: Packages specific to your project's needs, like specific rendering pipelines or custom tools.
  - **Built-In**
    - The Built-In section includes core Unity modules that are essential for the Unity Editor to function.
    - **Examples**: Core modules like the Input System, UI Toolkit, Physics.

Package Manager

- **Key Features:**

  - **Access to Packages**: for example, through the Unity Registry, and my Assets sections.
  - **Version Control**:
    - Easily switch between different versions of a package to ensure compatibility and stability.
    - Keep your project up-to-date with the latest features and bug fixes.
  - **Dependency Management**:
    - Automatically handle dependencies between packages, ensuring all required packages are installed.

- **Opening Package Manager**:
  - Navigate to Window > Package Manager in the Unity Editor.



- **Adding Packages**:
  - Select the desired registry (Unity Registry, My Assets, etc.).
  - Search for the package you need and click "Install."

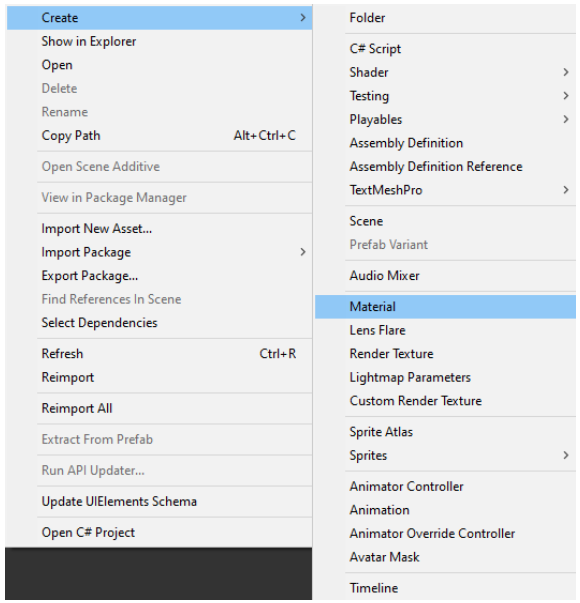
- **Updating and Removing Packages**:
  - **Update**: Select the installed package and click "Update" if a newer version is available.
  - **Remove**: Select the package and click "Remove" to uninstall it from your project.

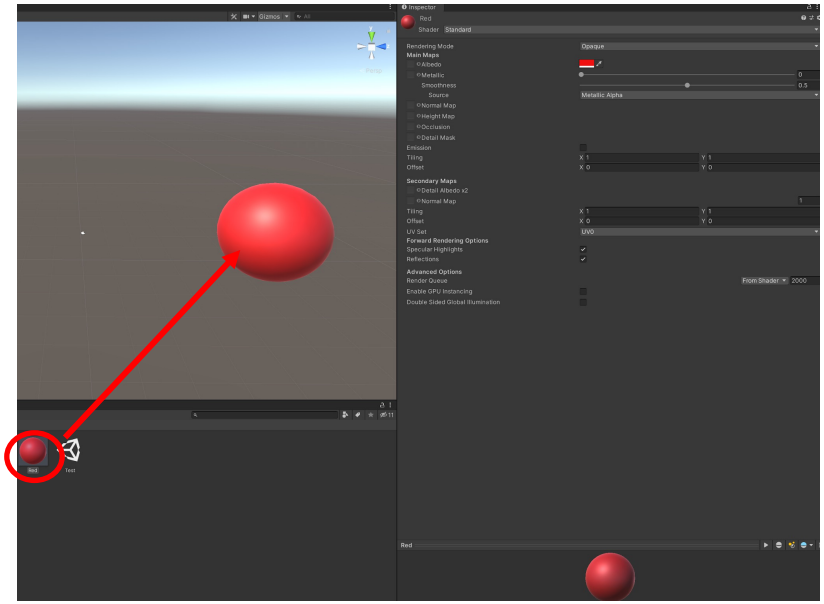
Work with materials (change the color of a Game Object):

We can give Game Objects different colors:

1. Right click on the **Assets** window:
2. Select **Create > Material**.
3. Click on the new created material and from the color slot of its inspector select the Color.
4. To apply the material, you can drag and drop the material on the object in the **Inspector window**, the **Scene window** or the **Hierarchy window**.

You can rename the resulting material by clicking on the name.

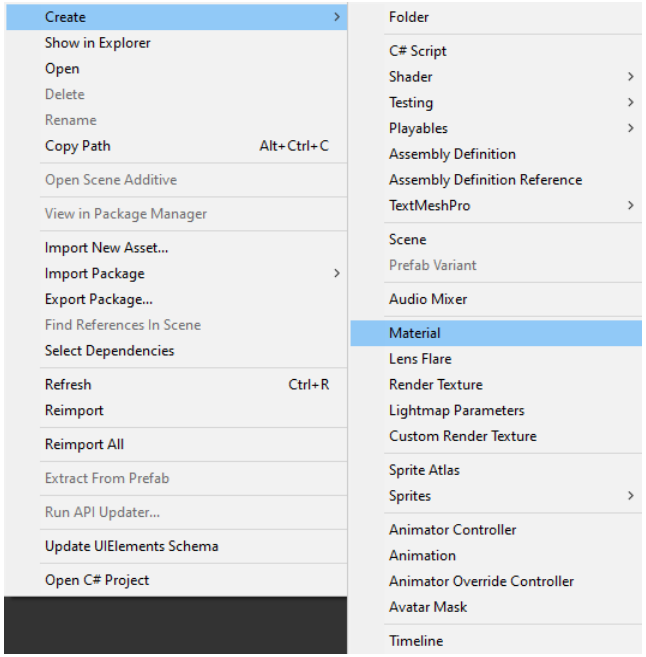


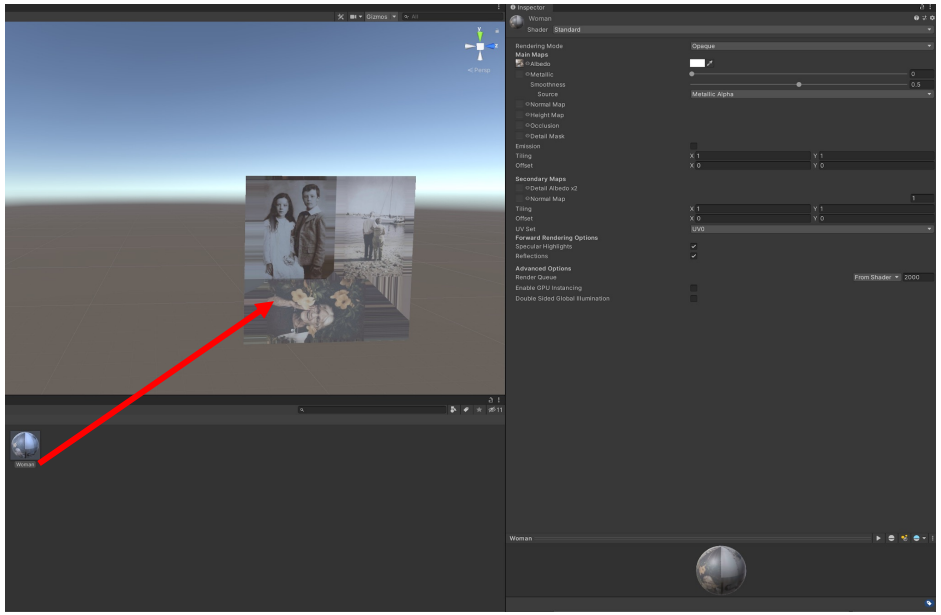


Work with materials (give them texture image):

We can create our own materials from texture images:

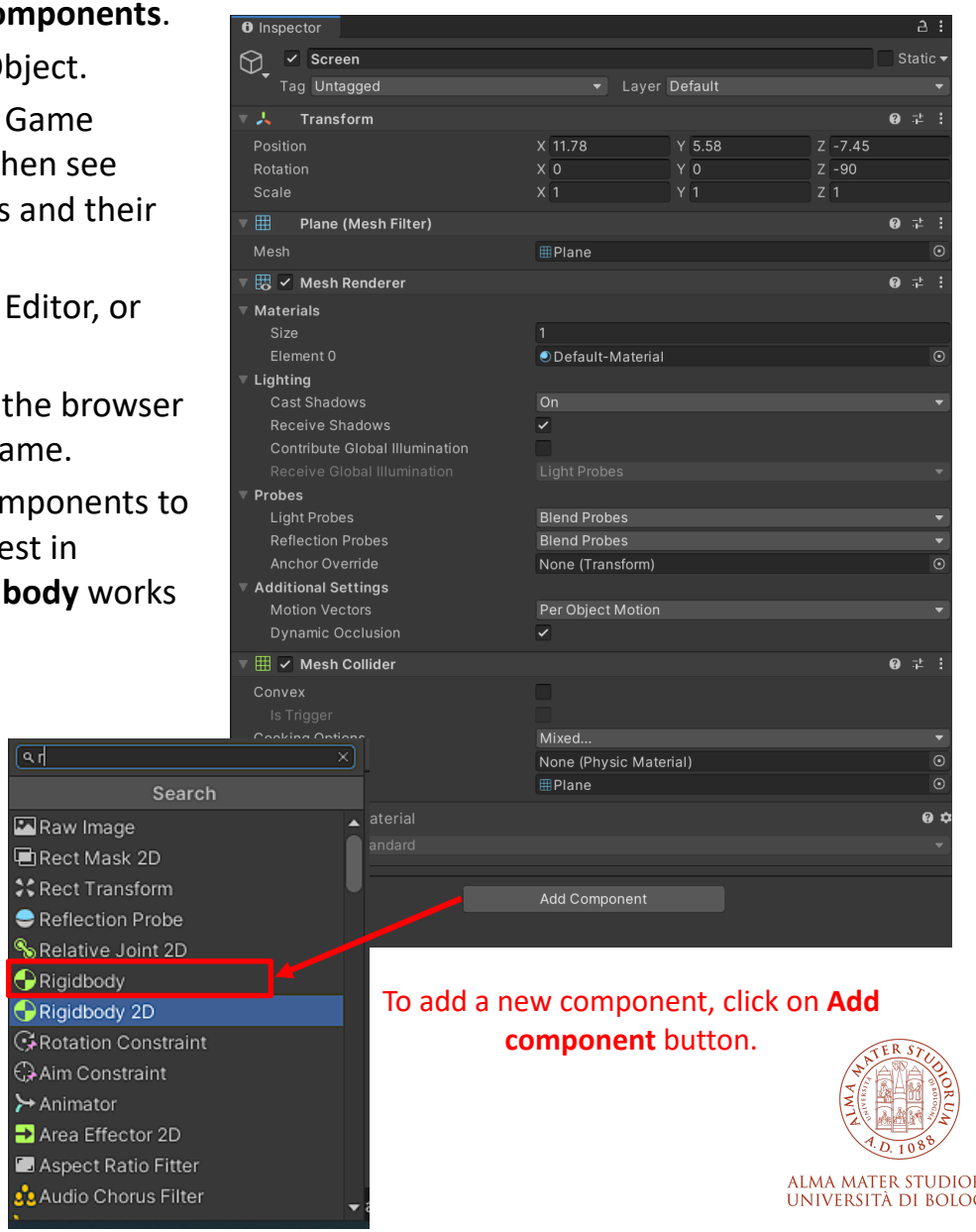
1. Right click in the **Assets** window.
2. Select **Create > Material**.
3. Import a texture image to your Unity (Choose a .jpg or .png square Image).
4. Click on the new created material and assign (drag and drop) the texture image to the **Albedo** slot of its **Inspector**.
5. To apply the material, you can drag and drop the material on the object in the **Inspector** window, the **Scene window** or the **Hierarchy window**.





Add a component to the game object:

- Each Game Object in the Unity Editor contains **components**.
- Components define the behavior of that Game Object.
- To view a Game Object's components, select the Game Object in the **Scene view** or **Hierarchy window**, then see the **Inspector window** for a list of all components and their Settings.
- You can interact with components directly in the Editor, or through script.
- You can navigate the components by category in the browser or use the search box to locate components by name.
- You can attach any number or combination of components to a single Game Object. Some components work best in combination with others. For example, the **Rigid body** works with a **Collider**.



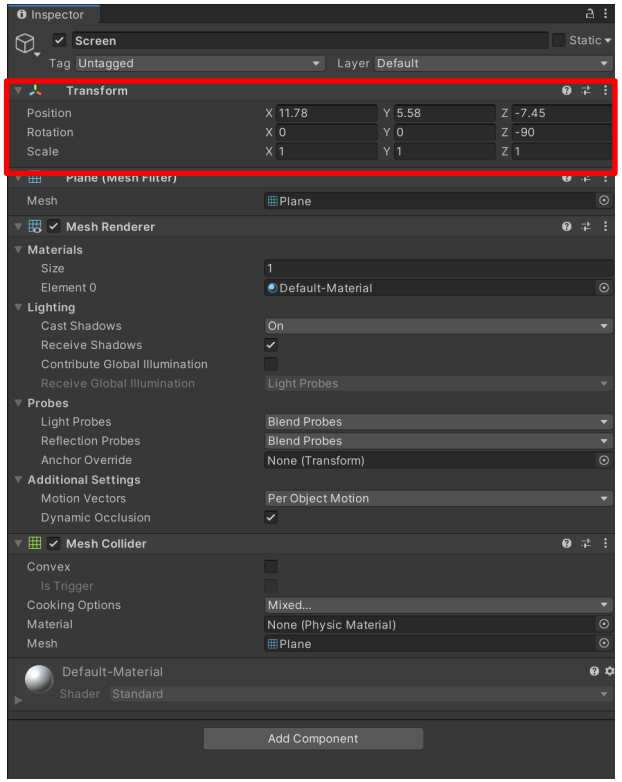
To add a new component, click on **Add component** button.

Introducing transform component:

- Every Game Object in Unity has a **Transform** component. It defines the Game Object's position, rotation, scale and is thus very important.
- A Game Object will always have a **Transform** component attached - it is not possible to remove a Transform or to create a Game Object without one.
- The position, rotation and scale values of a Transform are measured relative to the Transform's parent. If the Transform has no parent, the properties are measured in world space.

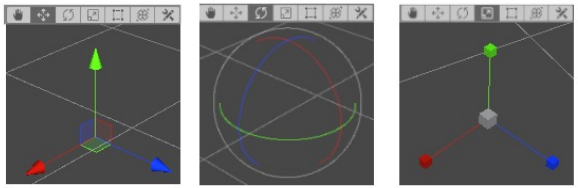
Properties

Property:	Function:
Position	Position of the Transform in X, Y, and Z coordinates.
Rotation	Rotation of the Transform around the X, Y, and Z axes, measured in degrees.
Scale	Scale of the Transform along X, Y, and Z axes. Value "1" is the original size (size at which the object was imported).

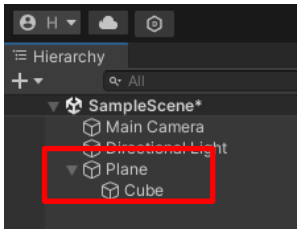


- A Transform can be edited in the **Scene View** or by changing its properties in the **Inspector**.
- In the scene, you can modify Transforms using the Move, Rotate and Scale tools.

These tools are located in the upper left-hand corner of the Unity Editor.



Relative Measurements:



Relative to the Parent

- **Hierarchy**:
  - In Unity, GameObjects can be organized in a parent-child hierarchy. A parent GameObject can have one or more child GameObjects.
- **Relative Transform**:
  - When a GameObject (child) is part of a hierarchy, its Transform values (position, rotation, and scale) are measured relative to its parent's Transform.
    - **Position**: The child's position is offset from the parent's position.
    - **Rotation**: The child's rotation is combined with the parent's rotation.
    - **Scale**: The child's scale is multiplied by the parent's scale.
- **Example**:
  - If a parent GameObject is at position (10, 0, 0) and the child GameObject is at position (2, 0, 0) relative to the parent, the child's world position would be (12, 0, 0).
  - If a parent GameObject has a rotation of (30, 0, 0) and the child GameObject has a local rotation of (10, 0, 0) relative to the parent, the child's world rotation would be (40, 0, 0).
  - If a parent GameObject has a scale of (2, 2, 2) and the child GameObject has a local scale of (1.5, 1.5, 1.5) relative to the parent, the child's world scale would be (3, 3, 3).

Introducing Collider components:

Colliders set rules around collisions for objects in the scene.

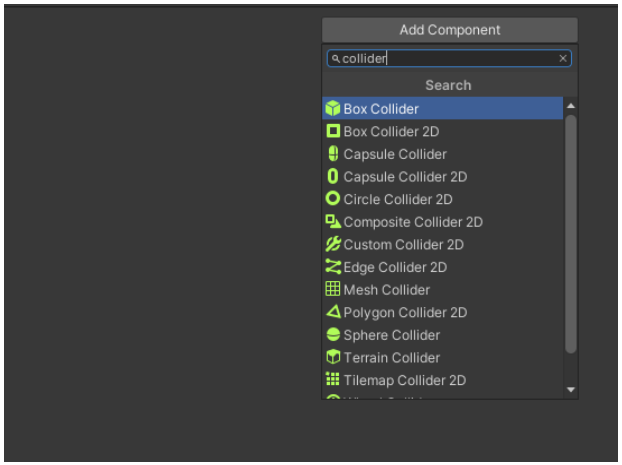
They ensure objects do not walk through each other.

Adding Colliders:

- Navigate to: Add Components -> Physics -> Colliders
- Multiple options available for different scenarios.

Types of Colliders:

- **Box Collider:** Commonly used on most objects.
- **Mesh Collider:** Adapts to the size and shape of the mesh; ideal for uneven shapes.
- **Sphere Collider:** Best for rounded objects.
- **Capsule Collider:** Typically used with characters.



Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- **Multimedia Integration and UI in Unity**
  - **Work with Audio**
  - **Work with Video**
  - **Work with UI and texts**
  - **Work with Animation**
- Working with the ProBuilder Tool in Unity
- Working with Lights
- Setting the Skybox

Work with audio:

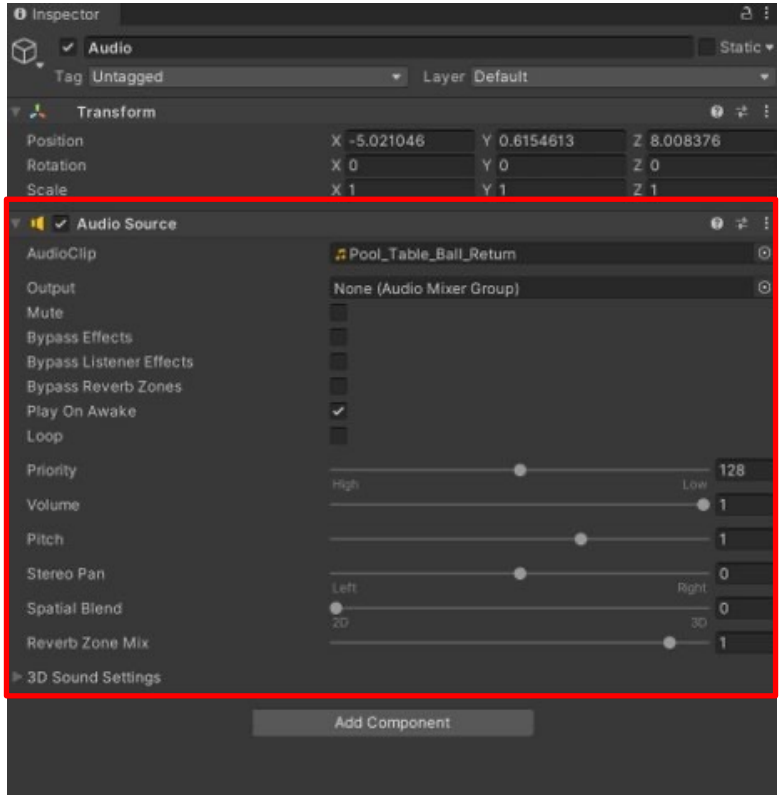
- To simulate the effects of position, Unity requires sounds to originate from **Audio Sources** attached to objects. The sounds emitted are then picked up by an **Audio Listener** attached to another object, most often the **Main Camera.**

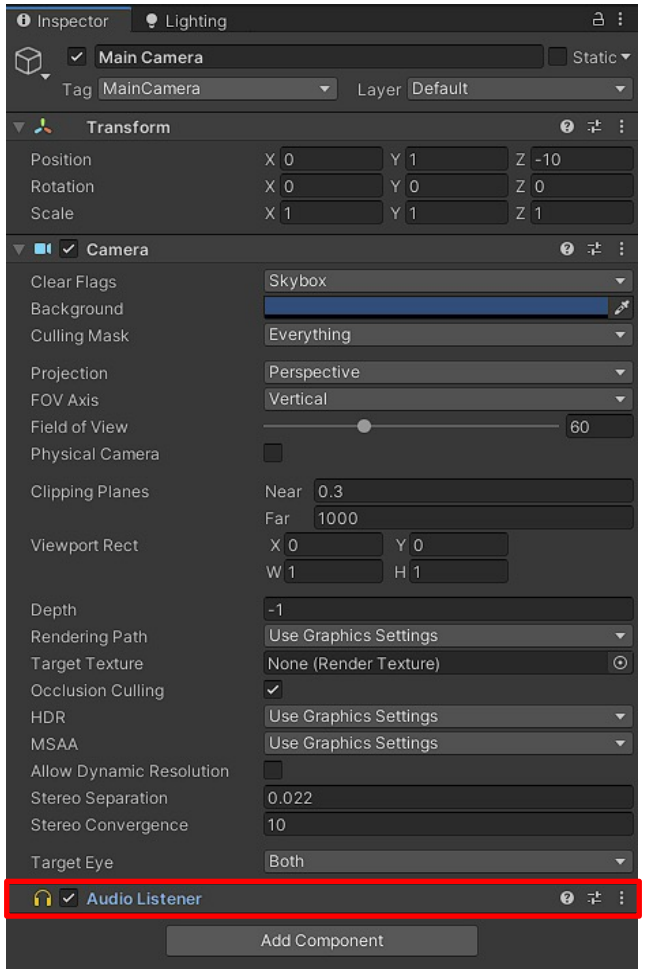
- Unity can then simulate the effects of a source's distance and position from the listener object and play them to the user accordingly.

- Unity can import audio files in **AIFF**, **WAV**, **MP3** and **Ogg** formats in the same way as other assets, simply by dragging the files into the **Project** panel. Importing an audio file creates an **Audio Clip** which can then be dragged to an **Audio Source** or used from a script.

Work with audio:

- **For a simply try on how to use it:**
  - Create an empty Game Object and name it, **Audio.**
  - Add the component of **Audio Source** to it.
  - Add an audio to the Asset folder of your unity.
  - Drag and drop it into the **Audio Clip** slot of the inspector.
  - Pay attention that an **Audio listener** component to be attached to the **Main Camera**.

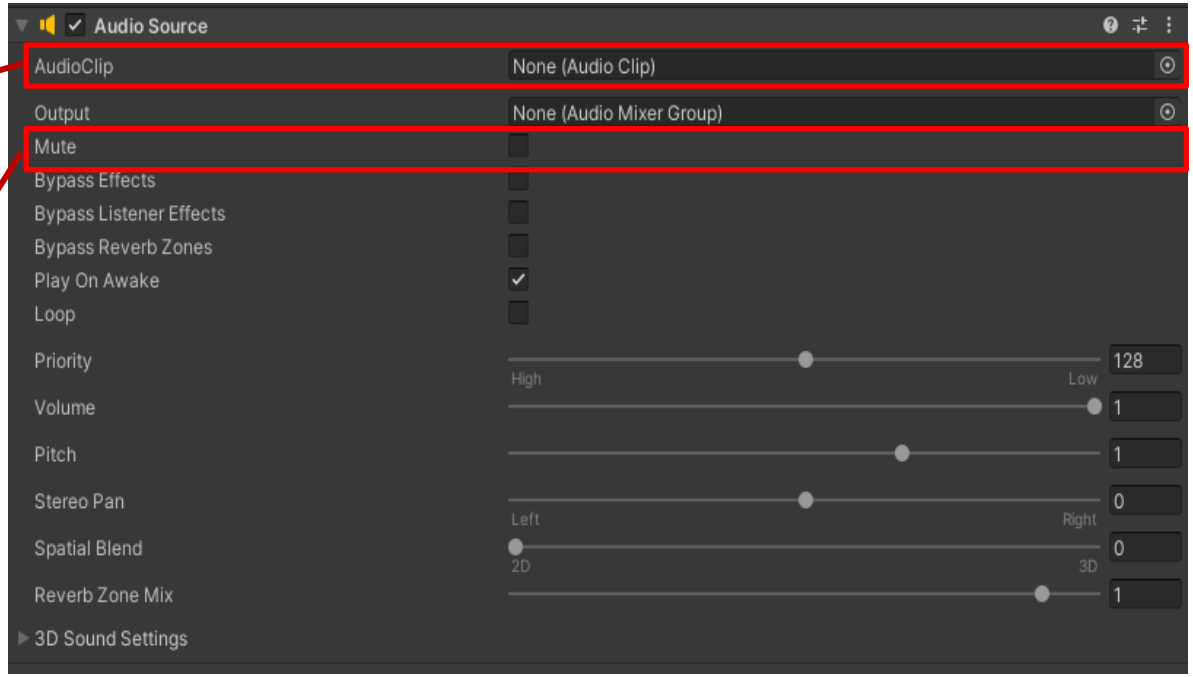




Work with audio:

It is used to assign audio files to the game object.

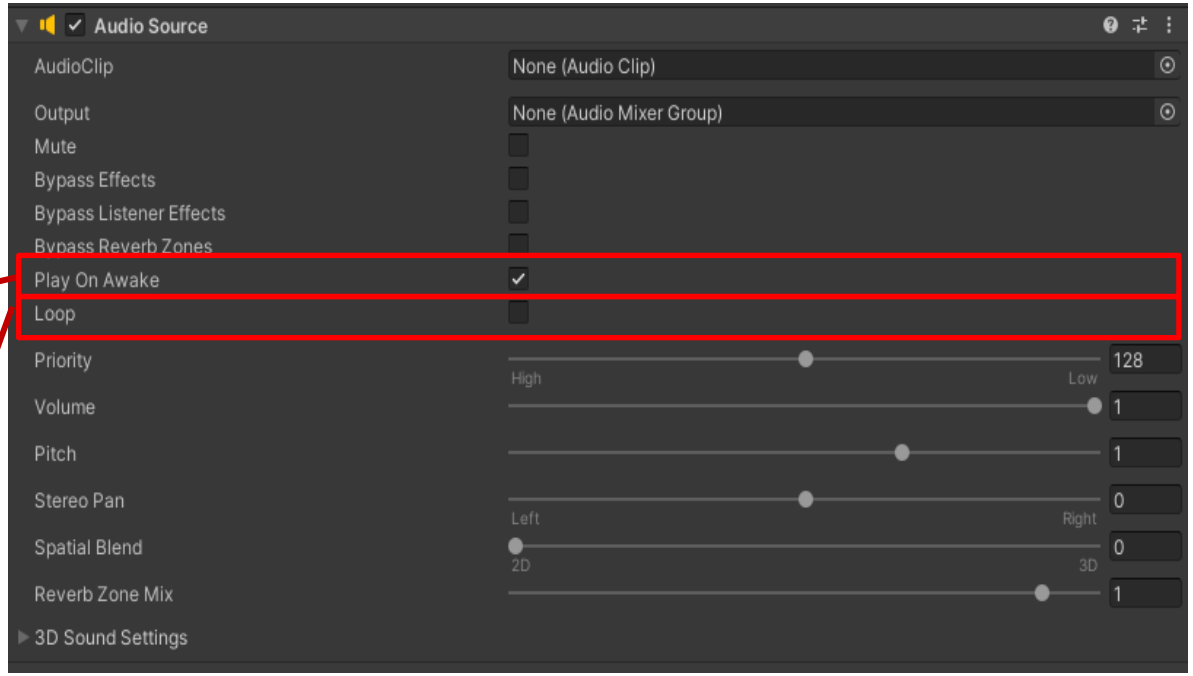
This option silences the audio, preventing it from playing any sound.



Work with audio:

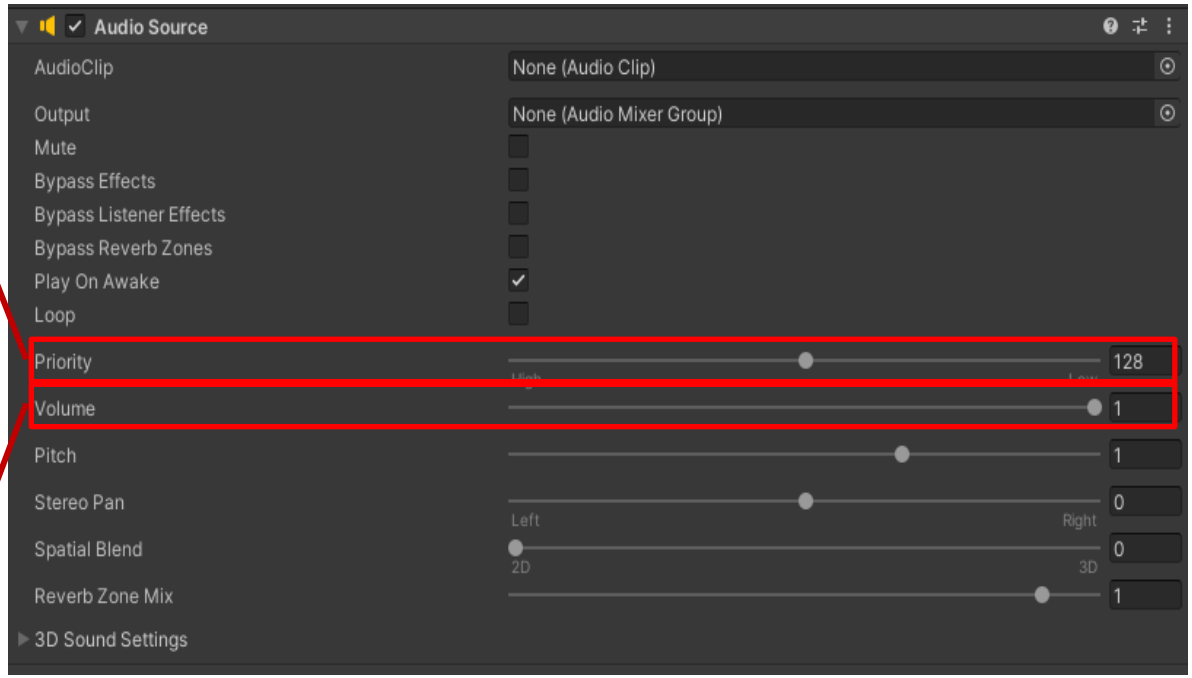
This option automatically starts the audio playback when the scene begins.

This option continuously repeats the audio clip.



Work with audio:

This option controls the priority of this sound relative to others.

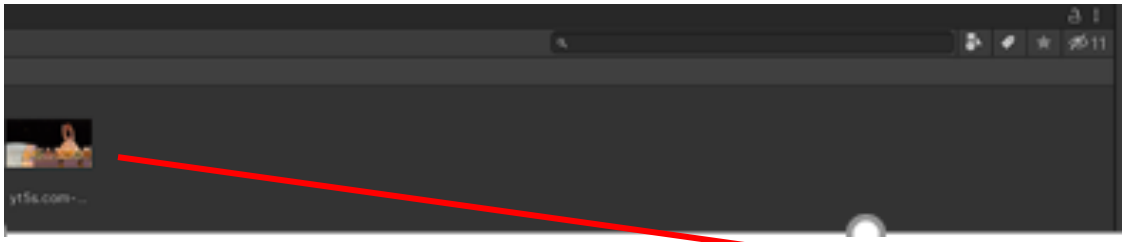


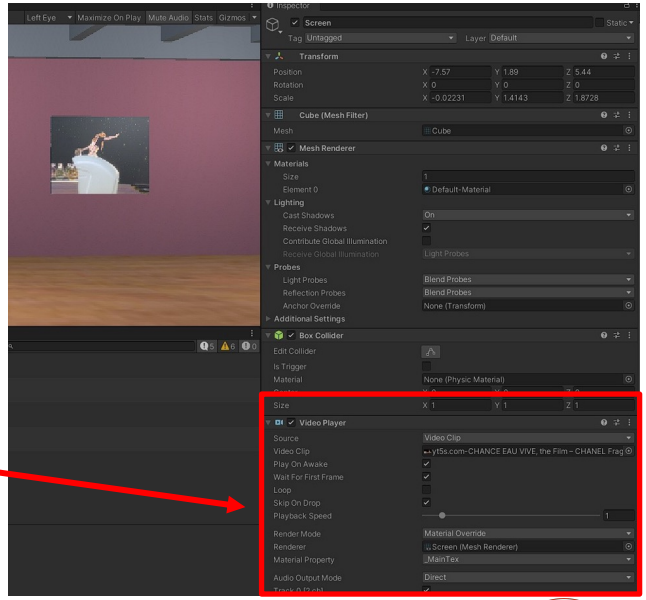
This property controls the loudness of the audio clip.

Work with Video:

- To use video in Unity, import **Video Clips** into Unity and configure them using the **Video Player** component. The system allows you to feed video footage directly into the **Texture** parameter of any component that has one. Unity then plays the Video on that Texture at run time.

- **For a simply try on how to use it:**
  - Create a **Cube** and name it, "Screen“ and set its scale and position.
  - Add the component of **Video Player** to it.
  - Add a **video** to the **Asset folder** of your unity.
  - Drag and drop it into the **Video Clip** slot of the inspector.





Work with UI: Canvas

Canvas is the area that all UI elements should be inside and must be children of it.

- **Canvas Creation:**

Creating a new UI element, such as an Image by right-clicking on the hierarchy > UI > Image, automatically creates a Canvas if there isn't already one in the scene.

- **Managing and Positioning UI Elements:**
  - The Canvas area is shown as a rectangle in the Scene View, making it easy to position UI elements without needing to have the Game View visible at all times.
  - UI elements are created as children of the Canvas.
  - UI elements in the Canvas are drawn in the same order they appear in the Hierarchy. The first child is drawn first, and the later children are drawn on top if they overlap.

- **EventSystem:**

Canvas uses the EventSystem object to help the Messaging System, managing user inputs and interactions.

Work with UI: Canvas

The Canvas has a **Render Mode** setting which can be used to make it rendered in **screen space** or **world space**.

Different types of Render Mode:

- **Screen Space - Overlay** places UI elements on the screen rendered on top of the scene. If the screen is resized or changes resolution, the Canvas will automatically change size to match this.

- **Screen Space - Camera** is similar to **Screen Space - Overlay**, but in this render mode the Canvas is placed in a given distance in front of a specified **camera**. The UI elements are rendered by this camera, which means that the camera settings affect the appearance of the UI. If the Camera is set to **Perspective**, the UI elements will be rendered with perspective, and the amount of perspective distortion can be controlled by the camera **Field of View**. If the screen is resized, changes resolution, or the camera frustum changes, the Canvas will automatically change size to match as well.

- **World Space:** in this render mode, the Canvas will behave as any other object in the scene. The size of the Canvas can be set manually using its **Rect Transform**, and UI elements will render in front of or behind other objects in the scene based on 3D placement. This is useful for UIs that are meant to be a part of the world.

Work with Rect tool and Rect transform:

The Rect Tool

- Every UI element is represented as a rectangle for layout purposes. This rectangle can be manipulated in the Scene View using the **Rect Tool** in the toolbar. The **Rect Tool** is used both for Unity's 2D features and for UI, and in fact can be used even for 3D objects as well.

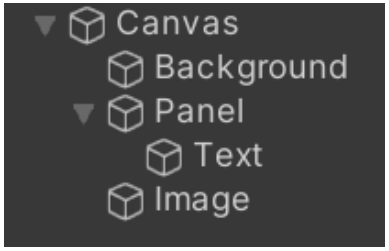
- The Rect Tool can be used to move, resize and rotate UI elements. Once you have selected a UI element, you can move it by clicking anywhere inside the rectangle and dragging. You can resize it by clicking on the edges or corners and dragging. The element can be rotated by hovering the cursor slightly away from the corners until the mouse cursor looks like a rotation symbol. You can then click and drag in either direction to rotate.

Rect Transform

- The **Rect Transform** is a new transform component that is used for all UI elements instead of the regular **Transform** component.
- **Rect Transforms** have position, rotation, and scale just like regular **Transforms**, but it also has a **width** and **height**, used to specify the dimensions of the rectangle.

Work with UI (Canvas, panel, text):

Goal: using **canvas**, **panel** and **text** to create a poster display.



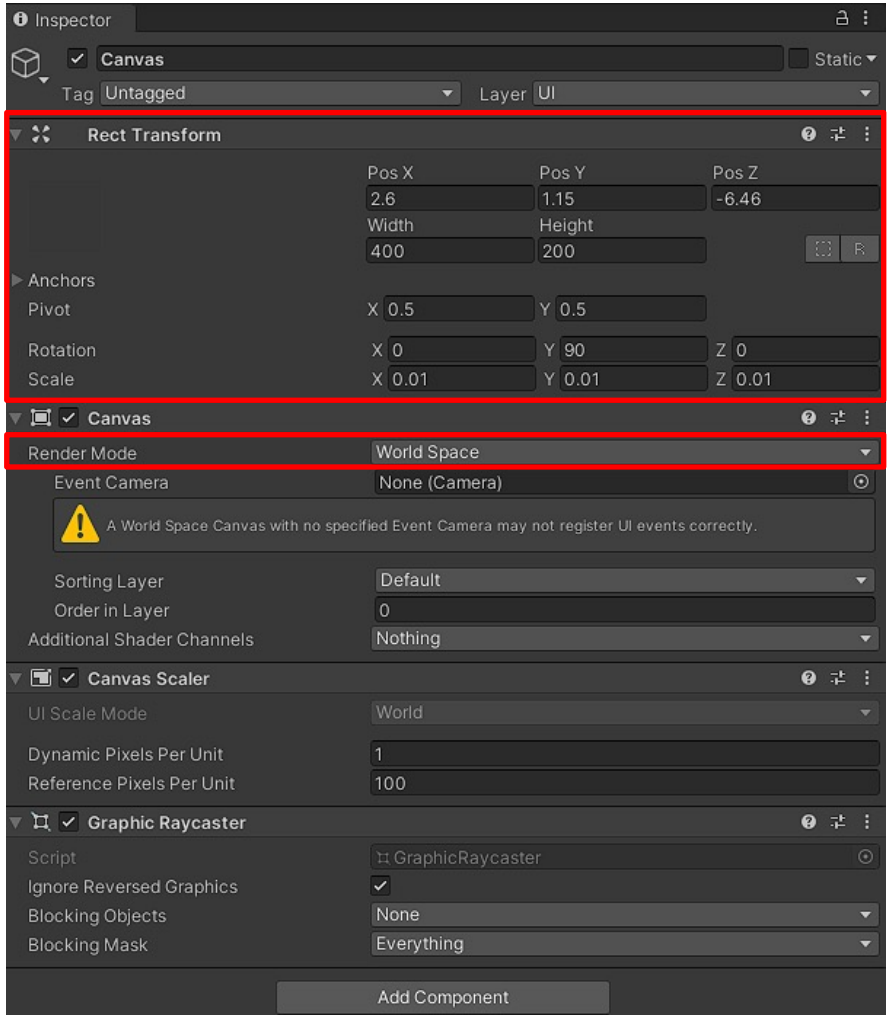
Work with UI (Canvas, panel, text):

Create a Canvas:

1. Right-click on the **Hierarchy -> UI -> Canvas**.
2. From its inspector change the **Render Mode** to **Word Space**.
3. In the **Rect Transfrom**, set its features.

For example, a proper settings could be with these changes:

Pos X = 2.6, **Pos Y** = 1.15, **Pos Z** = -6.46
Width = 400, **Height** = 200
Rotation = (0, 90, 0)
Scale = (0.01, 0.01, 0.01)



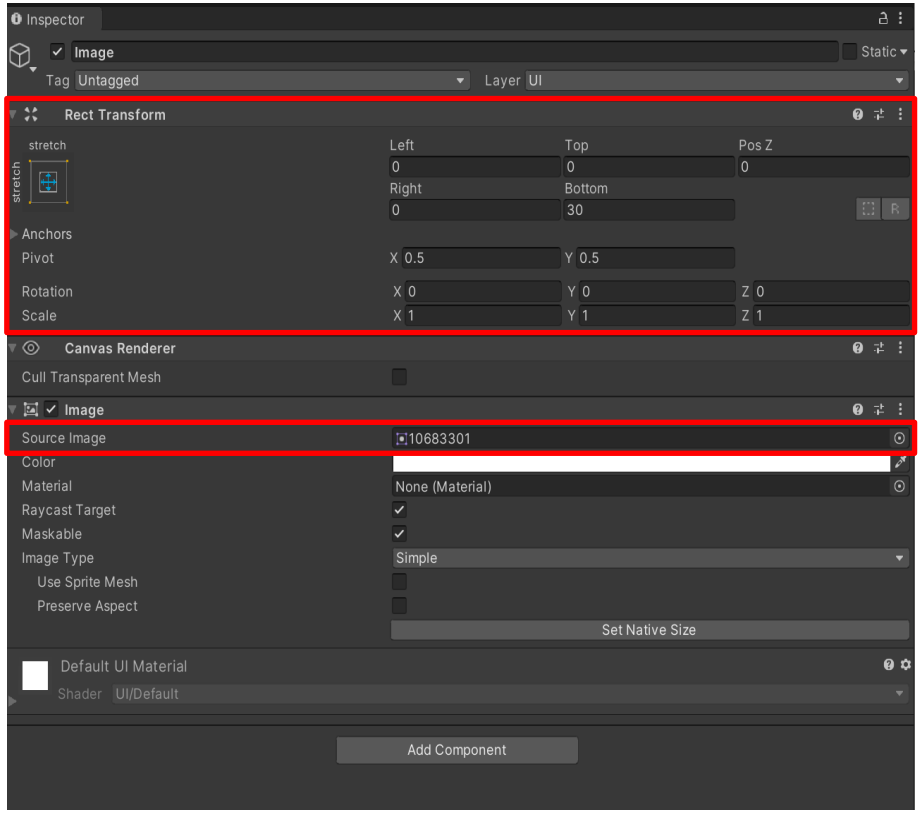
Work with UI (Canvas, panel, text):

Create an Image as the child of the Canvas:

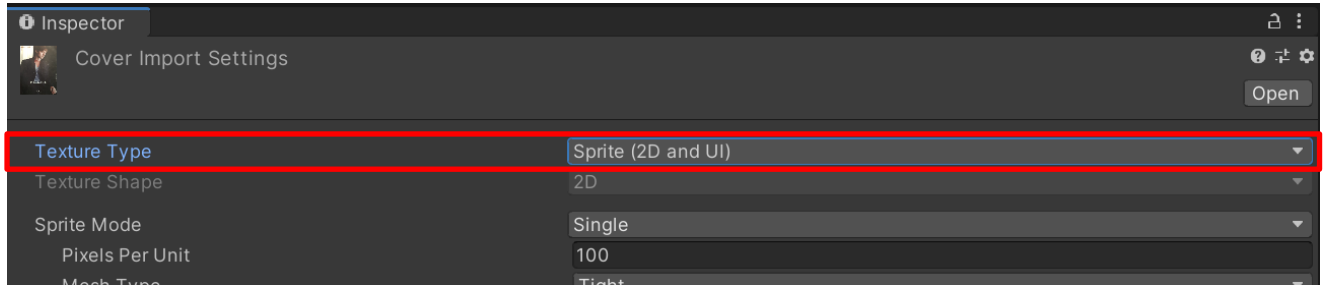
1. **Right-click on the Canvas in the Hierarchy -> UI -> Image.**
2. Then, in the **Rect Transfrom**, set its features.

For example, a proper settings could be with these changes:

Left = 0, **Top** = 0, **Pos Z** = 0
Right = 0, **Bottom** = 30



3. Add to the **Source Image** slot (by drag and drop) an image in the format of Sprite (2D). (To make an image as sprite 2d from its inspector, set the texture type to **Sprite 2D and UI**) and press **Apply**.



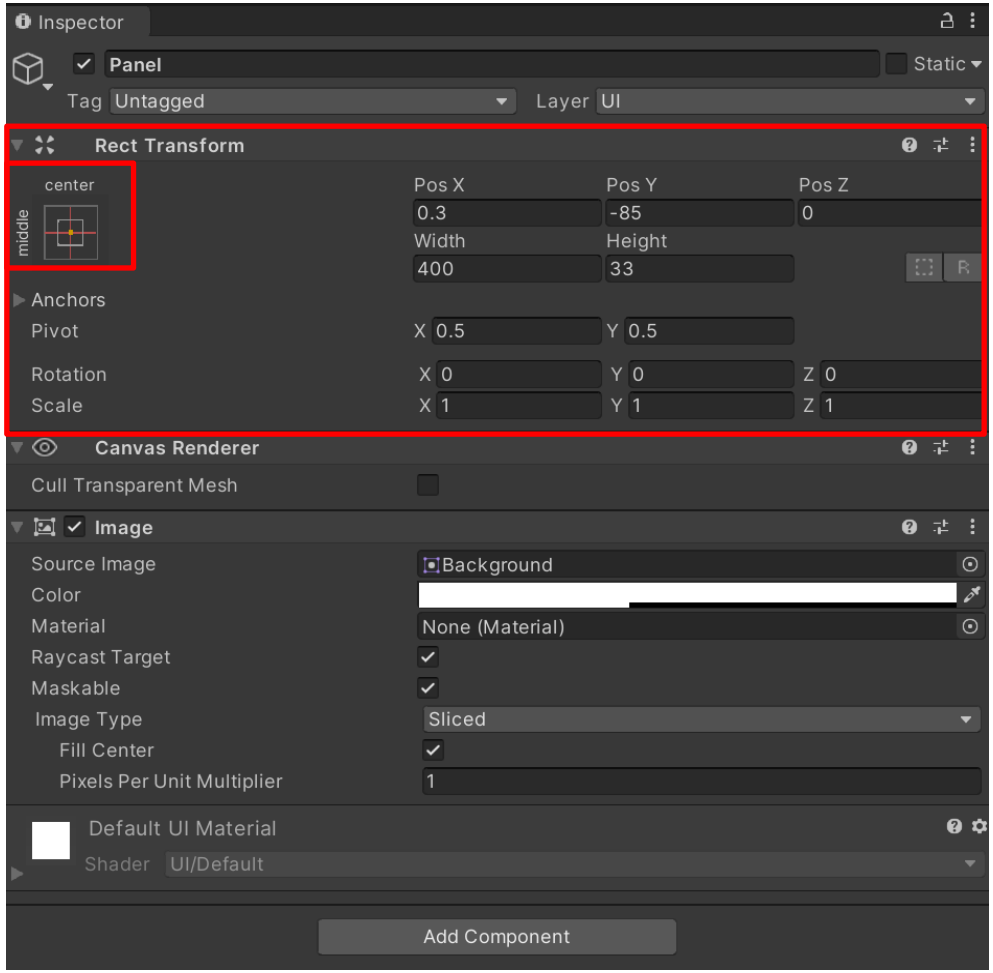
Work with UI (Canvas, panel, text):

Create a Panel as the child of the Canvas:

1. Right-click on the **Canvas** in the **Hierarchy -> UI -> Panel**.
1. Then, in the **Rect Transfrom**, set its features.

For example, a proper settings cou ld be this:

Pos X = 0.3, **Pos Y** = -85, **Pos Z** = 0 **Width** = 400, **Height** = 33



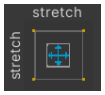
Work with UI (Canvas, panel, text):

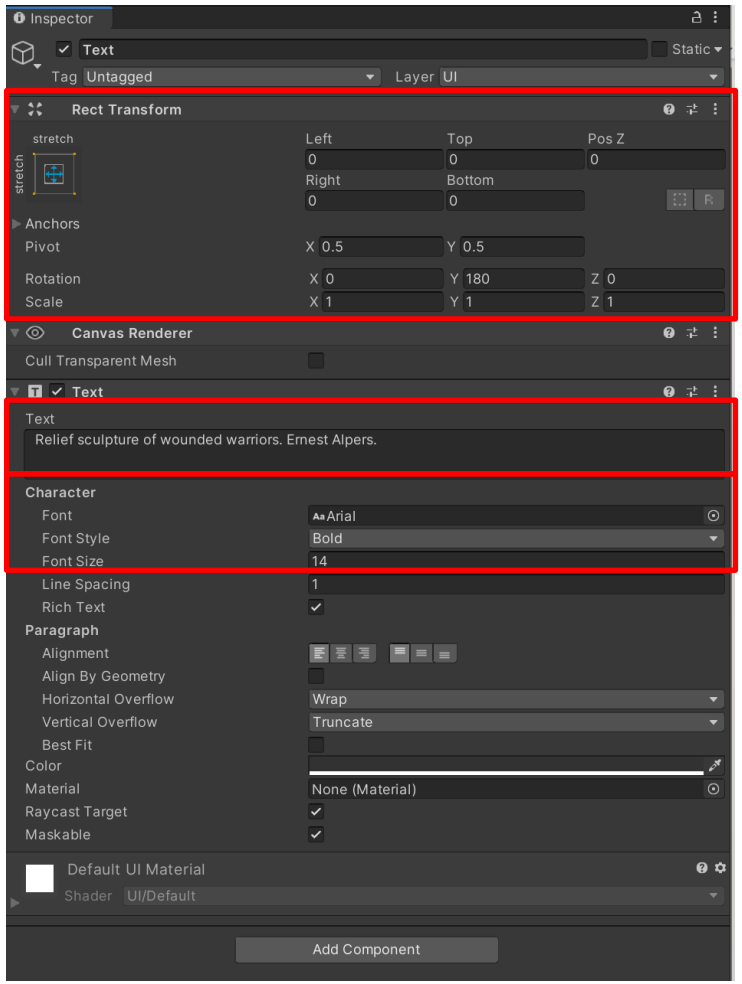
Create a Text as a child of the Panel:

1. Right click on the **Panel** in the **Hierarchy -> UI -> text**.
2. Then, in the **Rect Transfrom**, set its features.

For example, a proper settings could be with these changes:

Left = 0, **Top** = 0, **Pos Z** = 0
Right = 0, **Bottom** = 0
Rotation = (0, 180, 0)
Font: Arial
Font style: Bold
Font Size: 14
Write a text in the text slot.





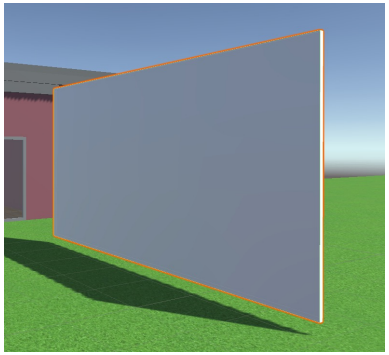
Create a background for the Canvas:

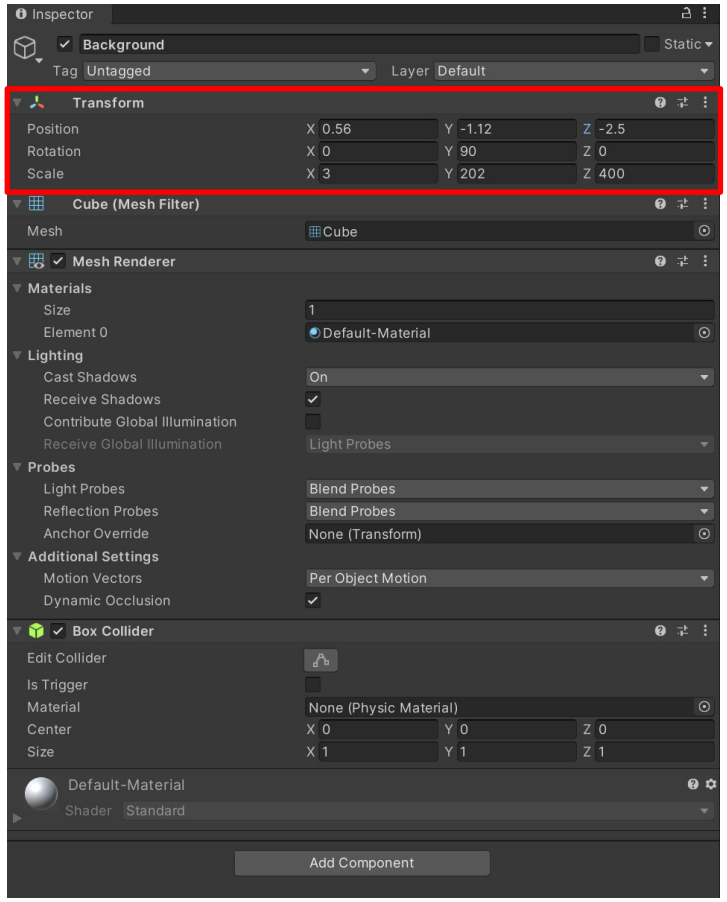
Create a background as a child of the Canvas:

- Create a Cube by right-clicking on the **hierrarchy -> 3D Object -> Cube**
- Make it as the child of the **Canvas**.
- Then, set its feature from the Inspector.

For example, a proper settings could be with these changes:

Position = (0.56, -1.12, -2.5)
Rotation = (0, 90, 0)
Scale = (3, 202, 400)





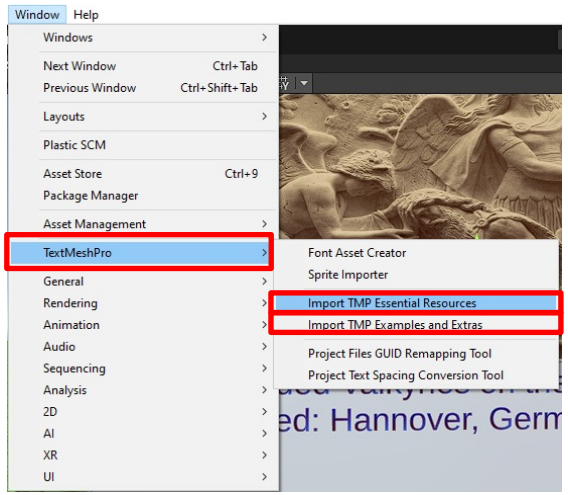
Required installations for working with TextMeshPro

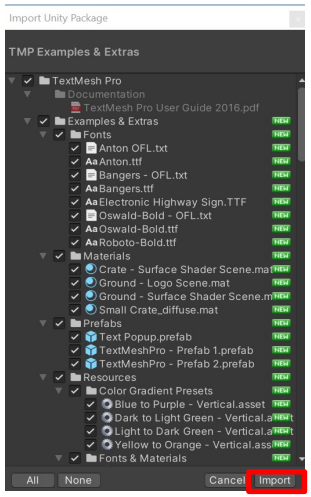
- The **TextMeshPro Package** is already included with the **Unity Editor** and as such does not require installation. To use that however we need to add some resources to our project: **TMP Essential Resources** and **TMP Examples and Extras**.

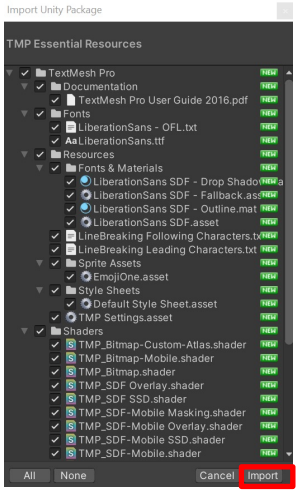
- To import the **TMP Essential Resources**, please use the menu “Window -> TextMeshPro -> Import TMP Essential Resources”. These resources will be added to your project in the **TextMeshPro folder.**

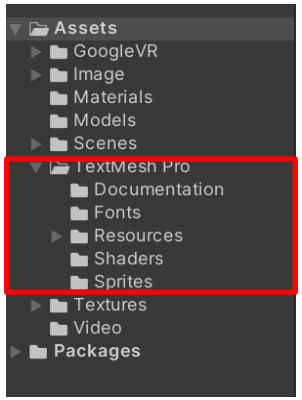
- The **TextMeshPro Package** also includes additional resources and examples that will make discovering and learning about **TextMeshPro**’s powerful features easier.

- To import the **TMP Examples & Extras**, please use the menu “Window -> TextMeshPro -> Import TMP Examples & Extras”. These resources will also be added in the same **TextMeshPro folder** inside your project.







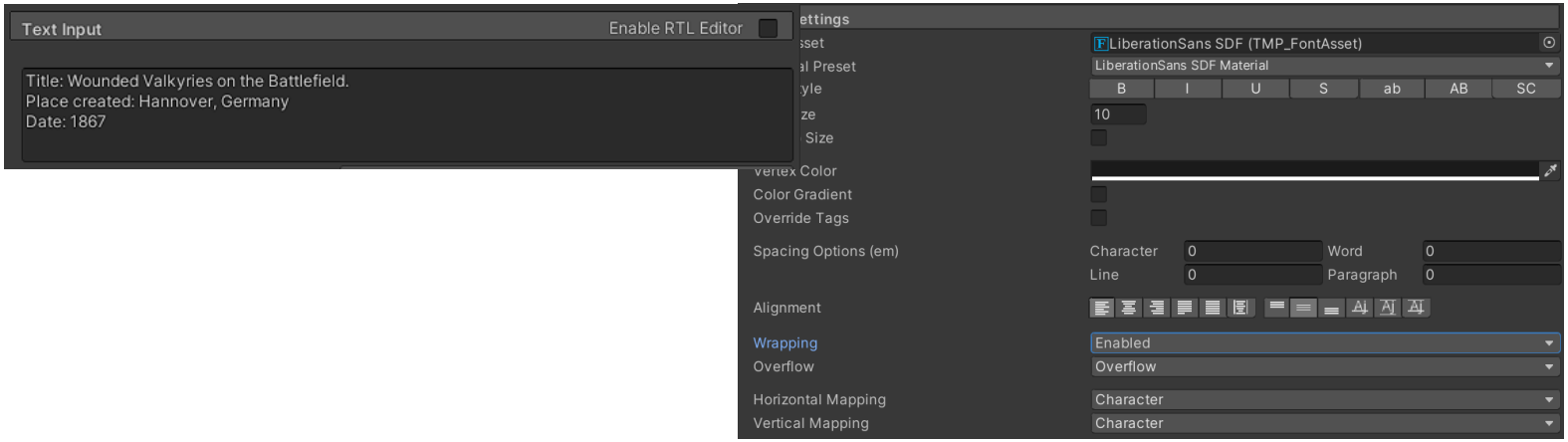


Add the TextMeshPro object to the scene:

- To add a **TextMeshPro object** to the scene:
  - If there is not a previous **Canvas** in your project, right-click on the **hierarchy** and use the menu "UI -> Text-TextMeshPro". A **Canvas** will be automatically created as a parent of the created **TextMeshPro object.**
  - If there is a previous **Canvas** in your project**,** you can create the **TextMeshPro object** as a child of the Canvas), by right-clicking on the name of it in the **hierarchy** and use the menu "UI-> Text-TextMeshPro".

- Each **TextMeshPro object** has a **TextMeshPro component**.
- Other objects can also have **TextMeshPro components**.

- **You can create some changes for the text through TextMeshPro component by these parts:**
  - **Text Input Field:** you can add **text** to the **TextMeshPro component**, by typing in this part.
  - **Main Settings part:** you can change the standard styling options from this part.



Let's play with some of the settings in the TextMeshPro component:

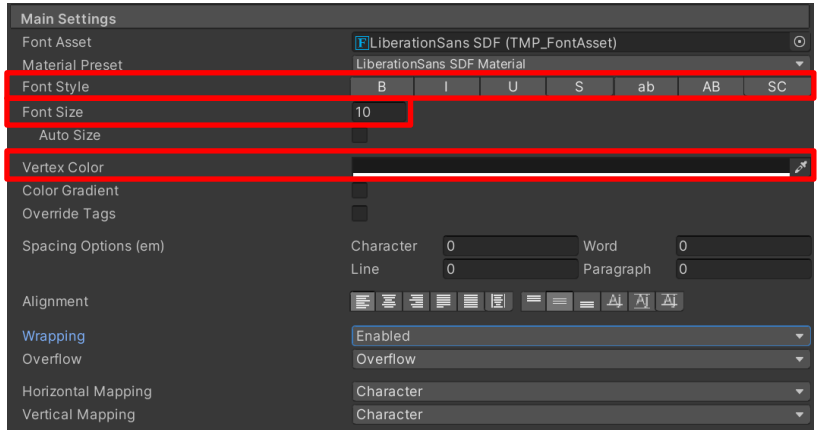
- **For example, let's start to edit the text of our previous example for image display:**

1. Put a multiple line text inside the Text Input part.

For example, "Title: Wounded Valkyries on the Battlefield
Place created: Hannover, Germany
Date: 1867"

2. Play with these options in the Main Settings and see what would happen:

- Font Style (bold, italic, underline, strikethrough, uppercase, lowercase, and smallcaps).
- Font Size
- Vertex Color

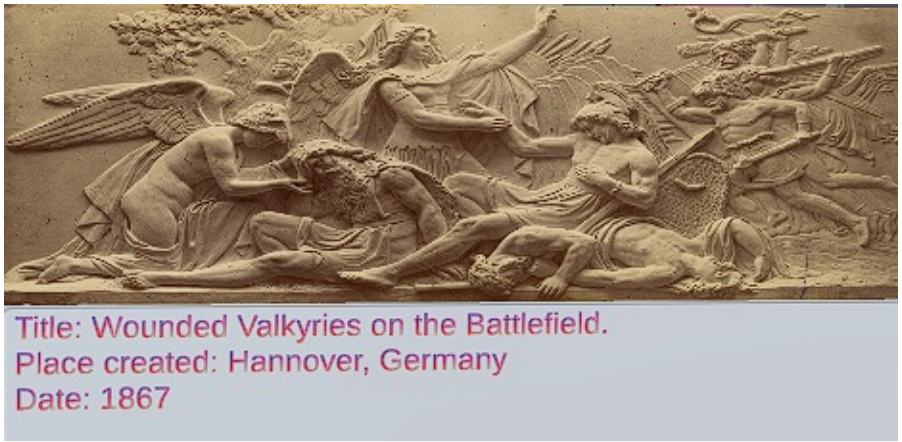


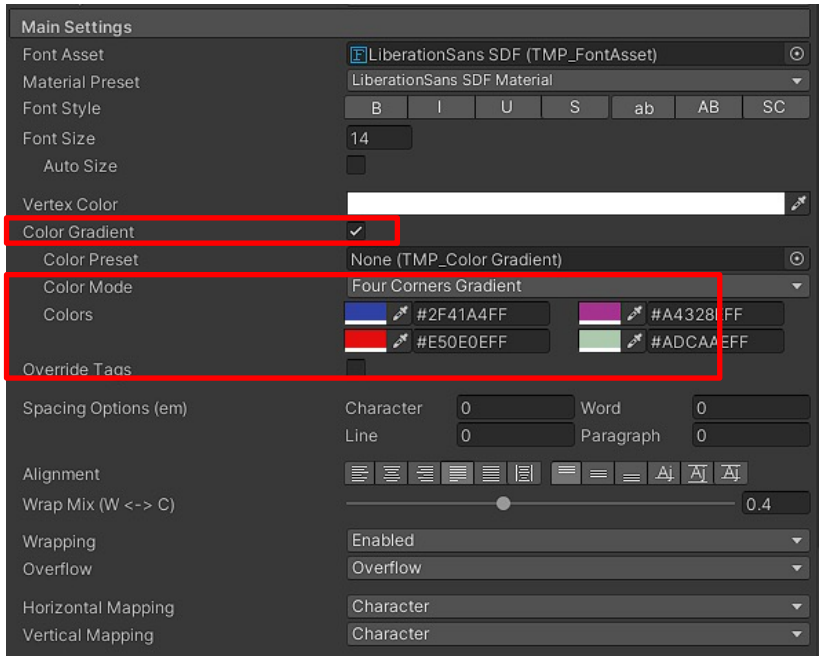
Let's play with some of the settings in the TextMeshPro component:

3. Work with Color Gradient:

- Enable the **Color Gradient** option.
- Change the **Color Mode** to **Four Corners Gradient**.
- Give different colors to the four corners.

(Pay attention: the **vertex color** should not be black if we want the applied color gradient be observable,)





Let's play with some of the settings in the TextMeshPro component:

4. Work with Rich Text tags:

- The styling options we tested all apply to the entire text block. However, there are many cases where you only need to apply styling to a portion of the text. Maybe you need one sentence, word, or even character to look different than the rest. To do this, you can use **Rich Text tags**.

- **Rich Text tags** work like **HTML tags:**
  - Angle brackets <> indicate an opening tag.
  - Angle brackets with a forward slash </> indicate a closing tag.
  - The tag affects all text between the opening and closing brackets.

Let's play with some of the settings in the TextMeshPro component:

4. Work with Rich Text tags:

- **Try to italicize a portion of the text:** in the **Text Input** field, place an italic opening tag **** before the target word and a closing tag **** after it.

- **Like this:**

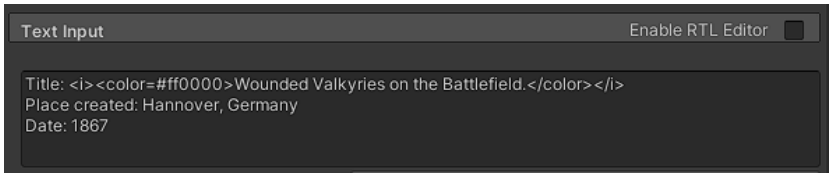
Let's play with some of the settings in the TextMeshPro component:

4. Work with Rich Text tags:

- **Try the color tag:** in the **Text Input** field, place **<color=red>** and **</color> tags** around target words or you can also specify the color by **hex code.**

  For example, red would use **<color=#ff0000>**.

- **Like this:**

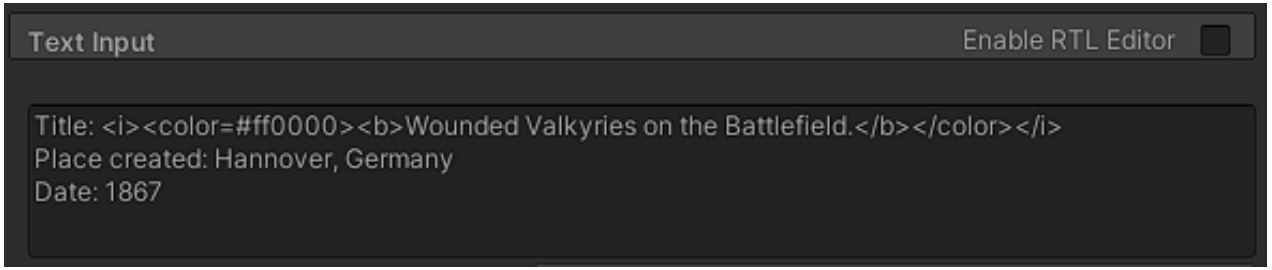


Or

Let's play with some of the settings in the TextMeshPro component:

4. Work with Rich Text tags:

- **Try the bold tag:** in the **Text Input** field, put the bold ** ** tags around the target words.
- **Like this:**



Let's play with some of the settings in the TextMeshPro component:

List of common color codes:

	White	#FFFFFF	rgb(255, 255, 255)		Lime	#00FF00	rgb(0, 255, 0)
	Silver	#C0C0C0	rgb(192, 192, 192)		Green	#008000	rgb(0, 128, 0)
	Gray	#808080	rgb(128, 128, 128)		Aqua	#00FFFF	rgb(0, 255, 255)
	Black	#000000	rgb(0, 0, 0)		Teal	#008080	rgb(0, 128, 128)
	Red	#FF0000	rgb(255, 0, 0)		Blue	#0000FF	rgb(0, 0, 255)
	Maroon	#800000	rgb(128, 0, 0)		Navy	#000080	rgb(0, 0, 128)
	Yellow	#FFFF00	rgb(255, 255, 0)		Fuchsia	#FF00FF	rgb(255, 0, 255)
	Olive	#808000	rgb(128, 128, 0)		Purple	#800080	rgb(128, 0, 128)

List of Basic style tags:

Bold	
Italics	
Underline	<u> </u>
~~Strikethrough~~	<s> </s>
Superscript - X^3 -	
Subscript - H_2O -	

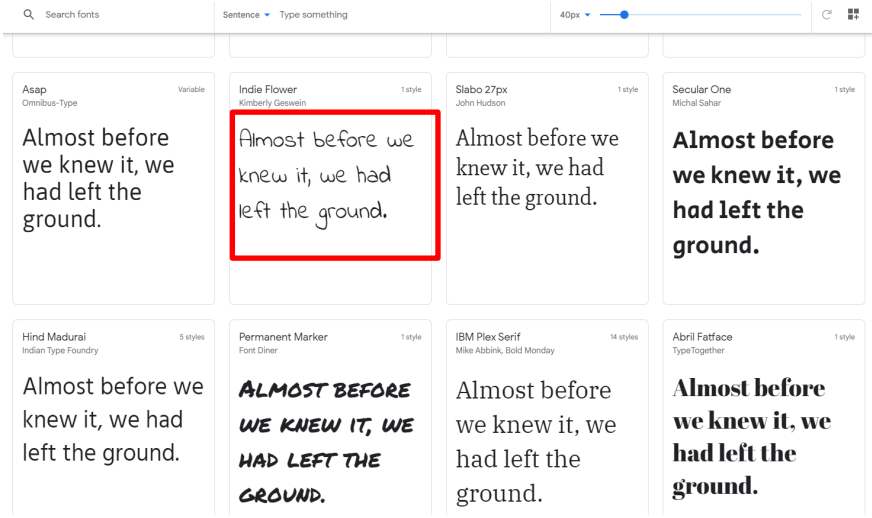
Let's play with some of the settings in the TextMeshPro component:

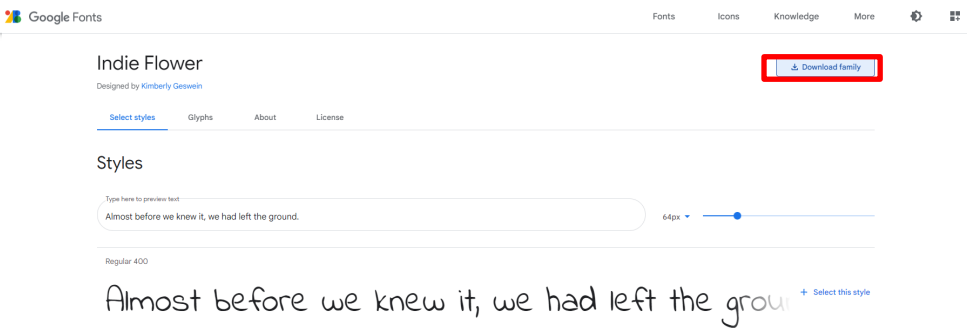
5. Create your own Font Asset:

- **Font Assets** are like containers for fonts. With them you can import your desired fonts into your project.

How to download:

1. Visit https://fonts.google.com/ , pick a font.
2. Click the **Download Family button** to download it.
3. A compact folder will be downloaded to your computer.



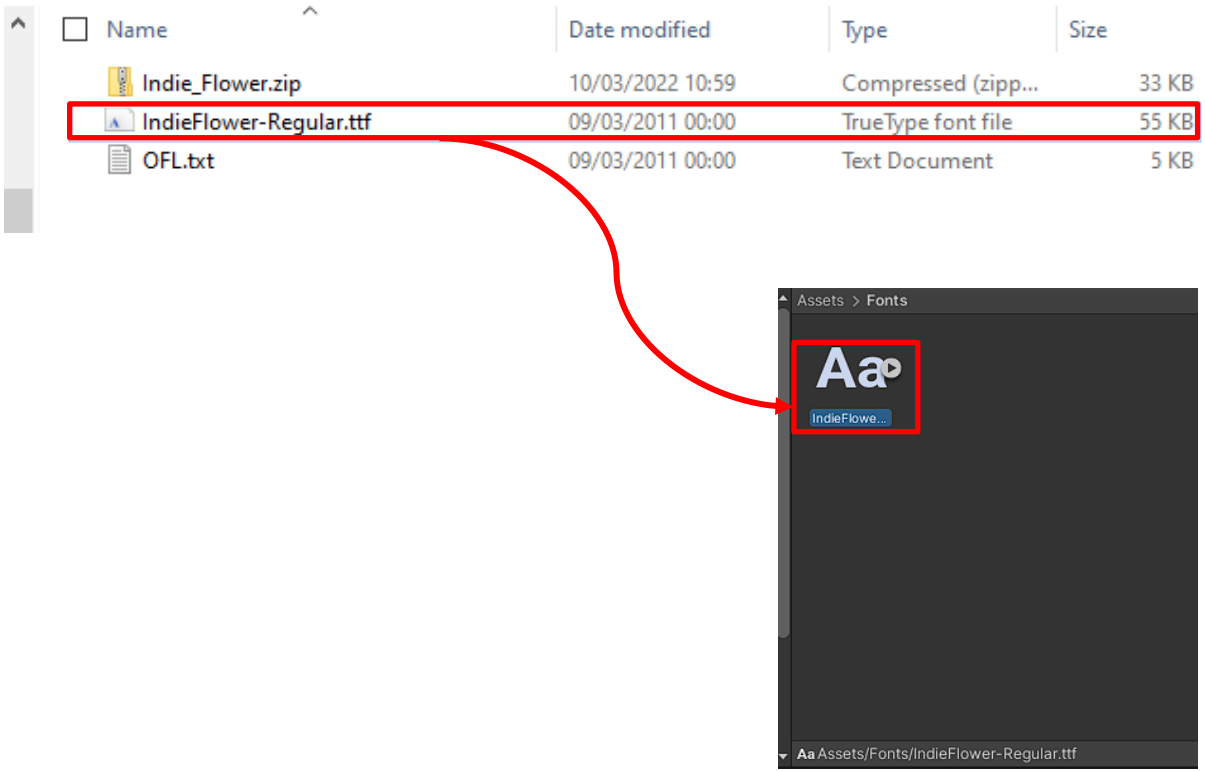


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Let's play with some of the settings in the TextMeshPro component:

How to set up Font Assets?

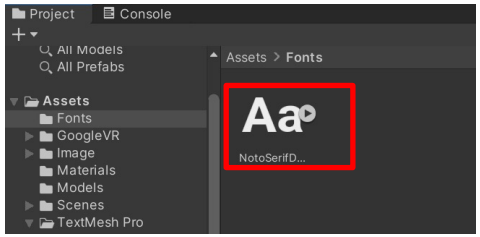
Extract the folder and import the **TTF (TrueType Font) file** into the **Unity project** (by dragging or using the menu "Assets -> Import New Asset ").

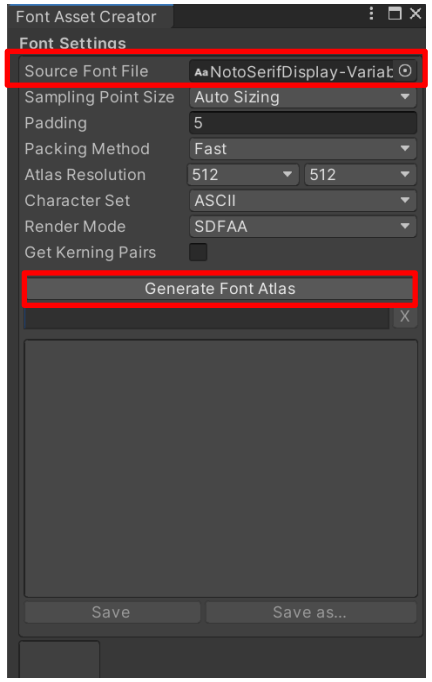


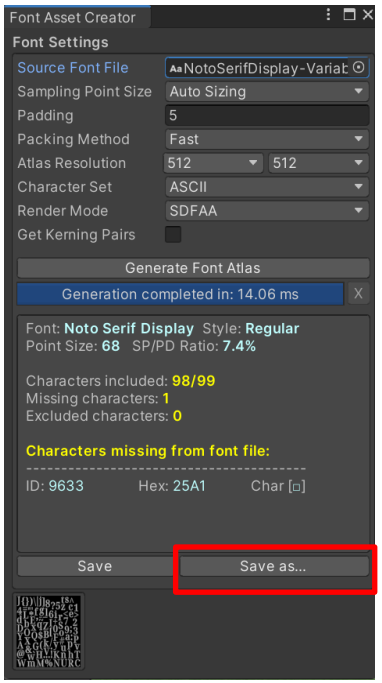
Let's play with some of the settings in the TextMeshPro component:

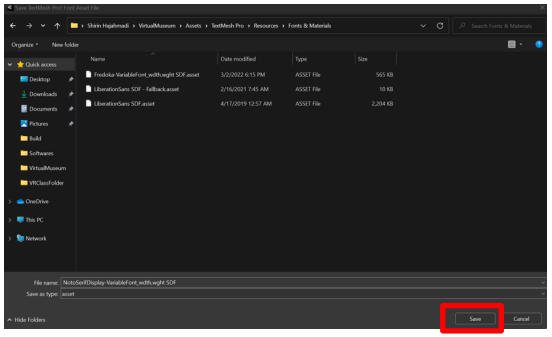
How to make the font usable by TextMeshPro (TMP)?

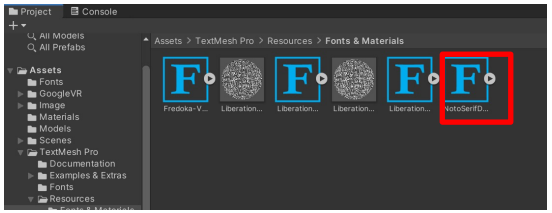
1. Use the menu "Window -> TextMeshPro -> select Font Asset Creator".
3. Drag the font you previously imported into the **Source Font File slot**.
4. Click on **Generate Font Atlas button**.
5. Click **Save as** and save it (the .asset file) in a folder of your project (e.g. fonts folder).







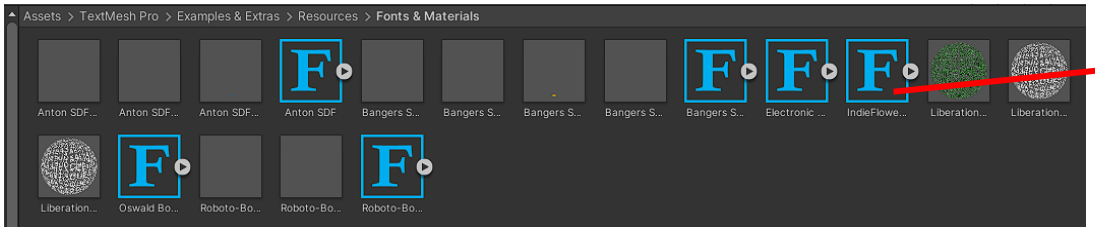


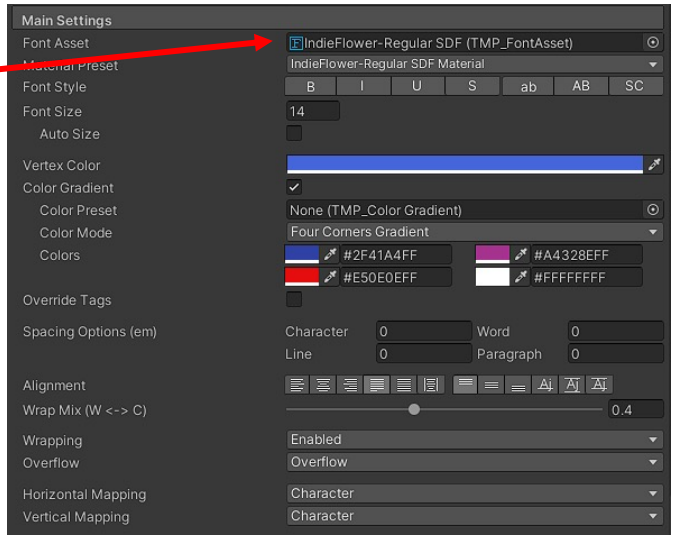


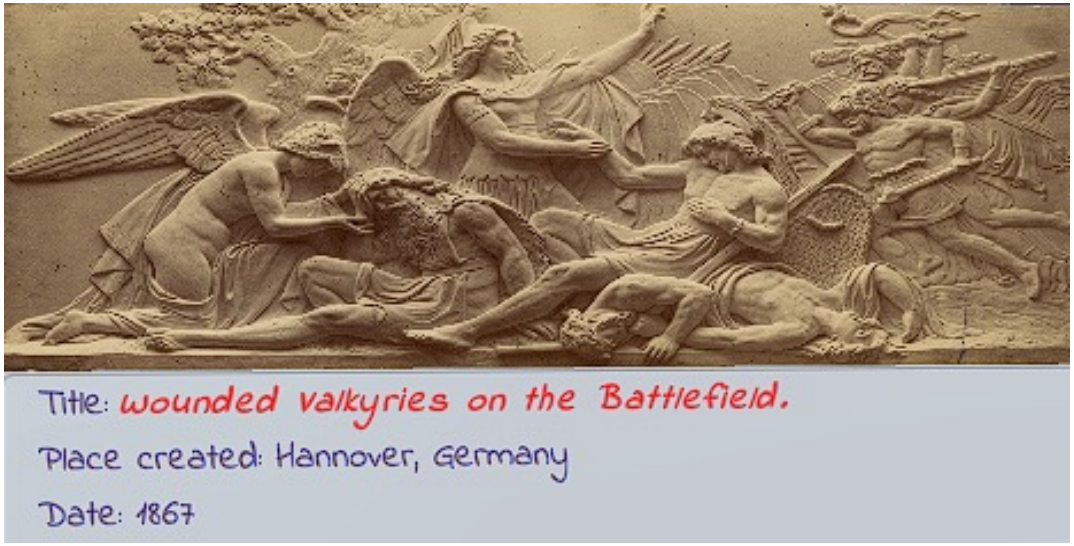
Set up Font Assets

How to give the selected text the newly created TMP Font Asset?

Drag the new created **font asset** into the **Font Asset field** of **TextMeshPro component**.







Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- **Working with the ProBuilder Tool in Unity**
- Working with Lights
- Setting the Skybox

Working with the ProBuilder Tool in Unity:

Adding ProBuilder Tool through Package Manager in Unity

Open the Package Manager:

1. Go to the top menu in Unity and select **Window > Package Manager**.

Find ProBuilder:

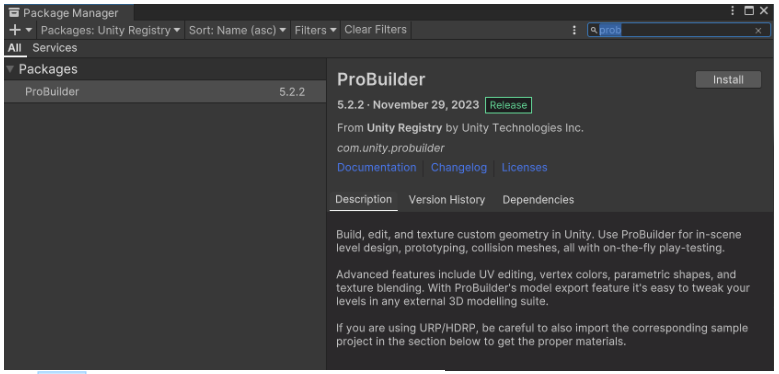
1. In the Package Manager window, switch to the **Unity Registry** tab.
2. Scroll through the list or use the search bar to find **ProBuilder**.

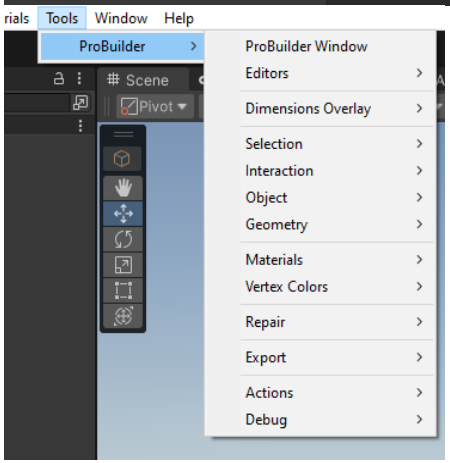
Install ProBuilder:

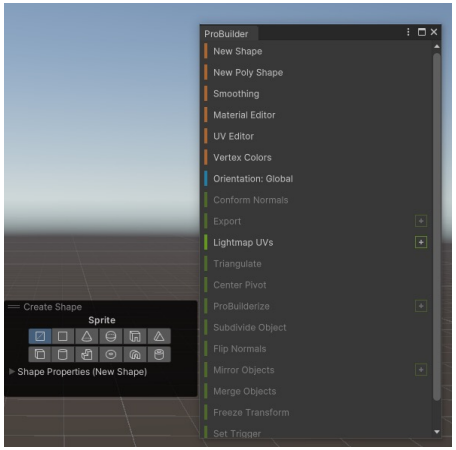
1. Select **ProBuilder** from the list.
2. Click the **Install** button on the bottom right of the Package Manager window.

Access ProBuilder:

1. After installation, you can access the ProBuilder tools.
2. Go to the top menu and select **Tools > ProBuilder Window**.





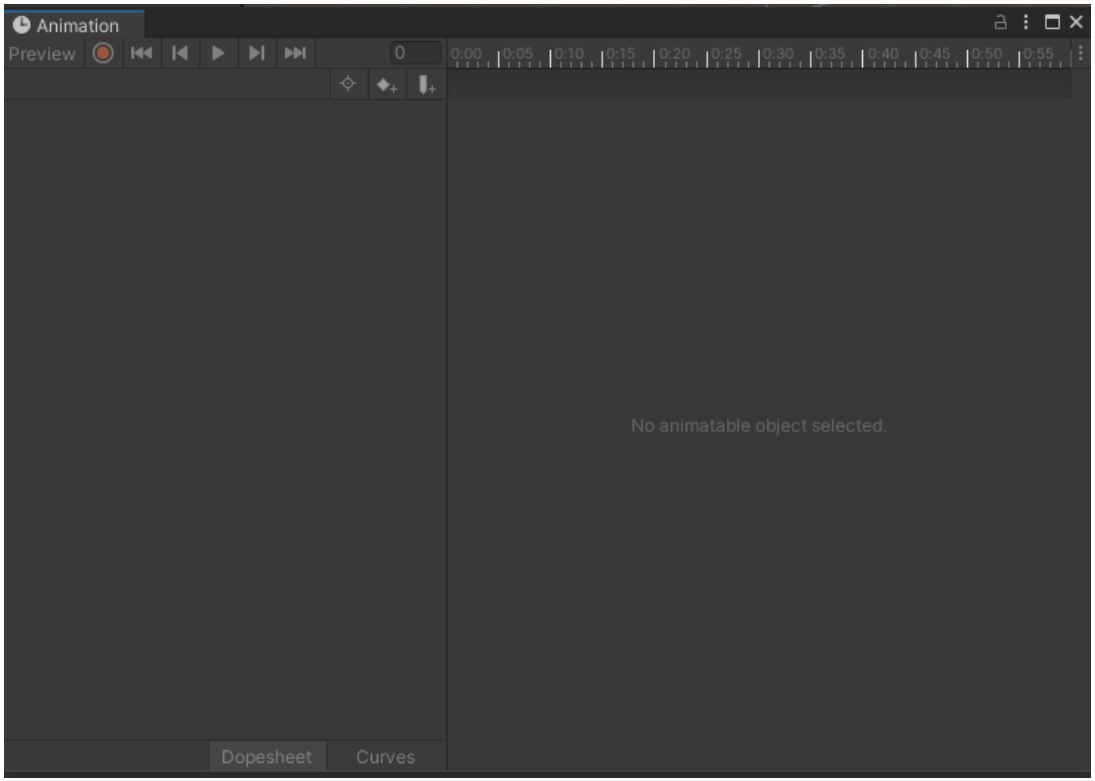


Creating animations through the Animation Window:

The Animation Window allows you to create and edit animations for Game Objects within Unity.

Accessing the Animation Window:

- Go to **Window > Animation > Animation** to open the Animation Window.

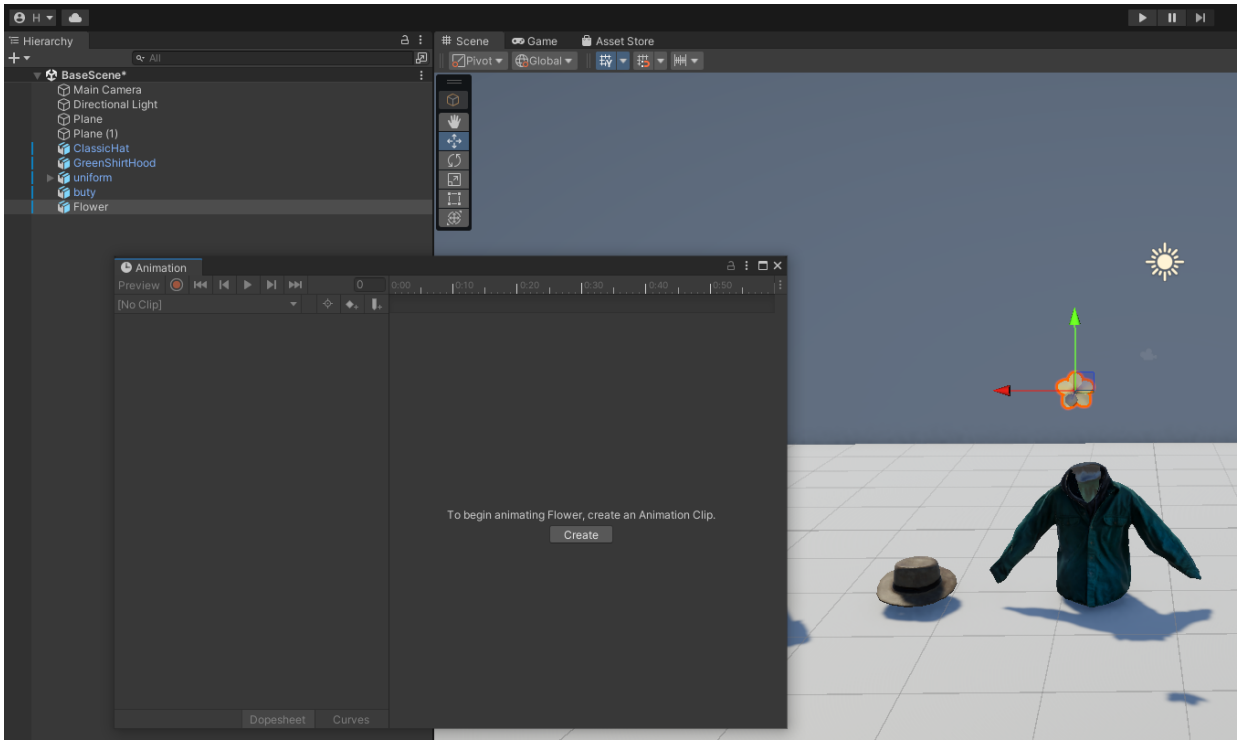


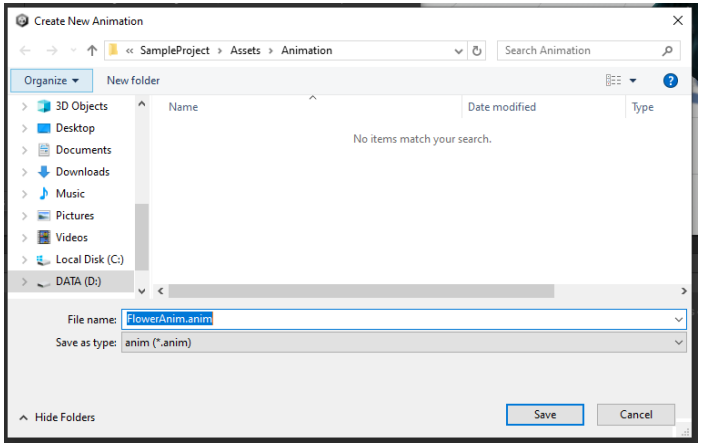
Setting Up for Animation:

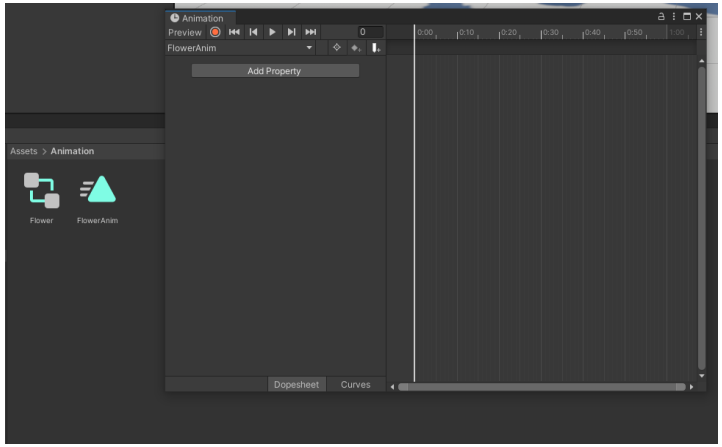
Creating a New Animation Clip:

1. Click on the Game Object in the Hierarchy that you want to animate.
2. If the Game Object doesn't have an Animator component, Unity will create one.
3. Create a folder named "Animation" inside the project's Assets folder.
4. In the Animation Window, click the **Create** button to create a new Animation Clip.
5. Name your Animation Clip and save it in your Animation folder.
6. The timeline at the bottom of the Animation Window shows the duration of your animation.
7. Click the **Red Circle** button to enter Record Mode.
8. Move the playhead to different points in the timeline and modify the Game Object's properties (position, rotation, scale, etc.).
9. Unity will automatically create keyframes at these points.
10. Press the **Play** button in the Animation Window to preview your animation.
11. The Animator component on your Game Object will now use this Animation Clip.
12. Ensure your Animation Clip is saved and applied to the Game Object.

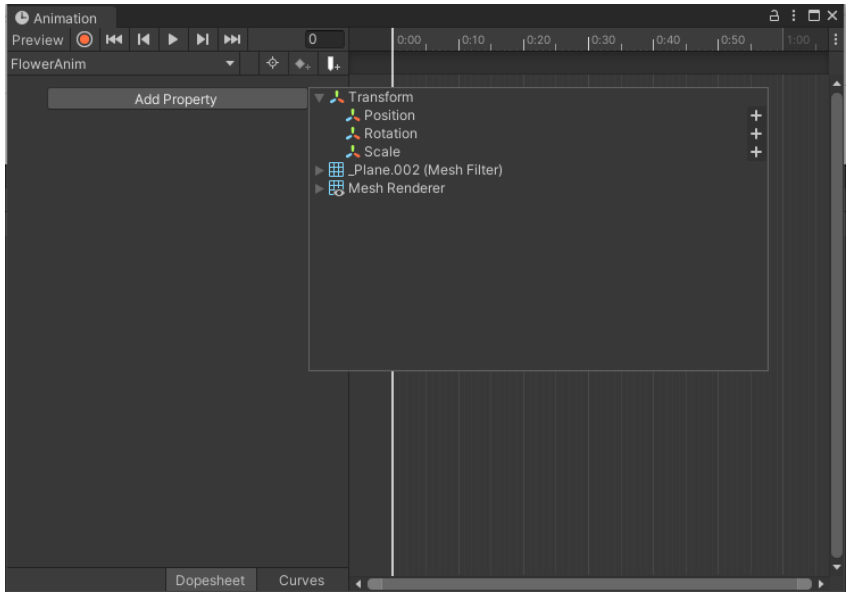
Setting Up for Animation:

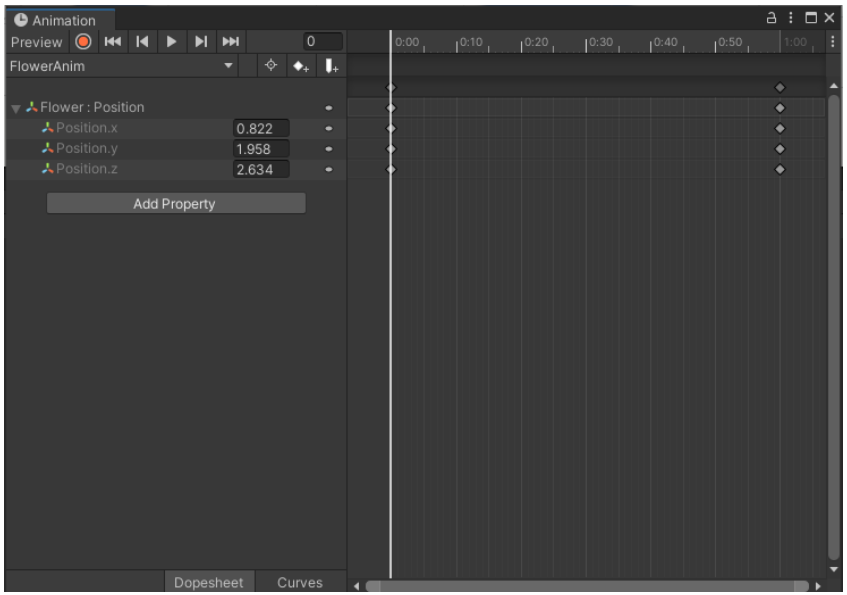






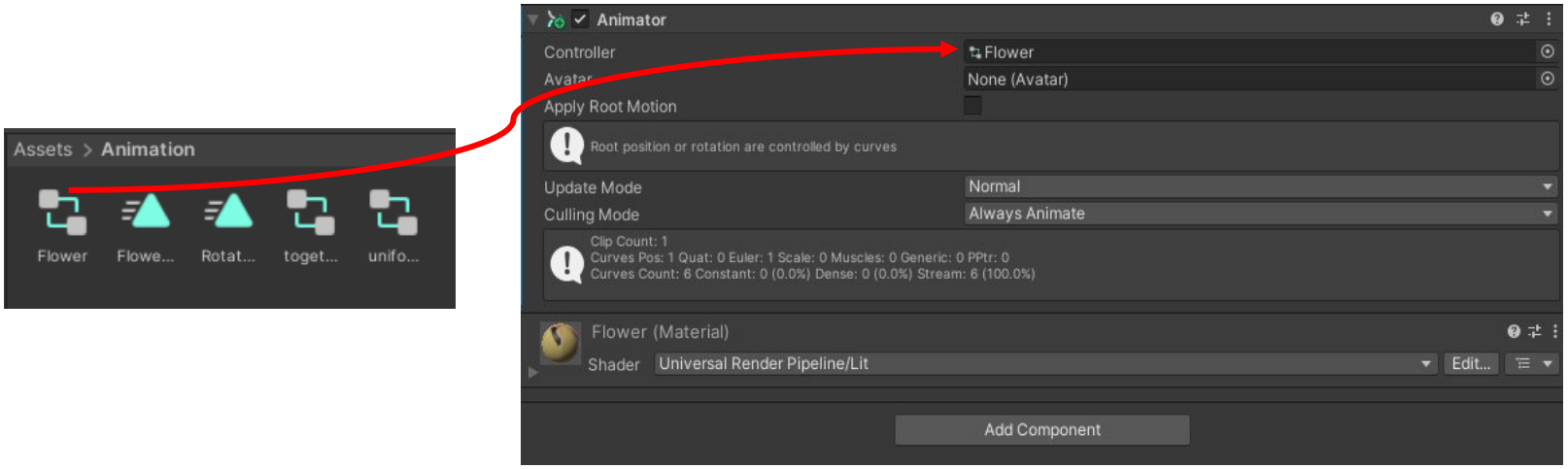
Setting Up for Animation:





Animation Controllers

- The Animation Controller manages multiple animation clips and controls the transitions between them.
- Allows for complex animation sequences and blending based on game logic.
- **Creating an Animation Controller**
  - Right-click in the Project window and select **Create > Animator Controller** or let the Animation Window create one automatically if it does not already exist.
  - Name your new Animator Controller and save it in your Animation folder.
- **Assigning the Animator Controller**
  - Select the Game Object in the Hierarchy that you want to animate.
  - In the Inspector, drag your Animator Controller into the **Controller** field of the Animator component of the selected Game Object.



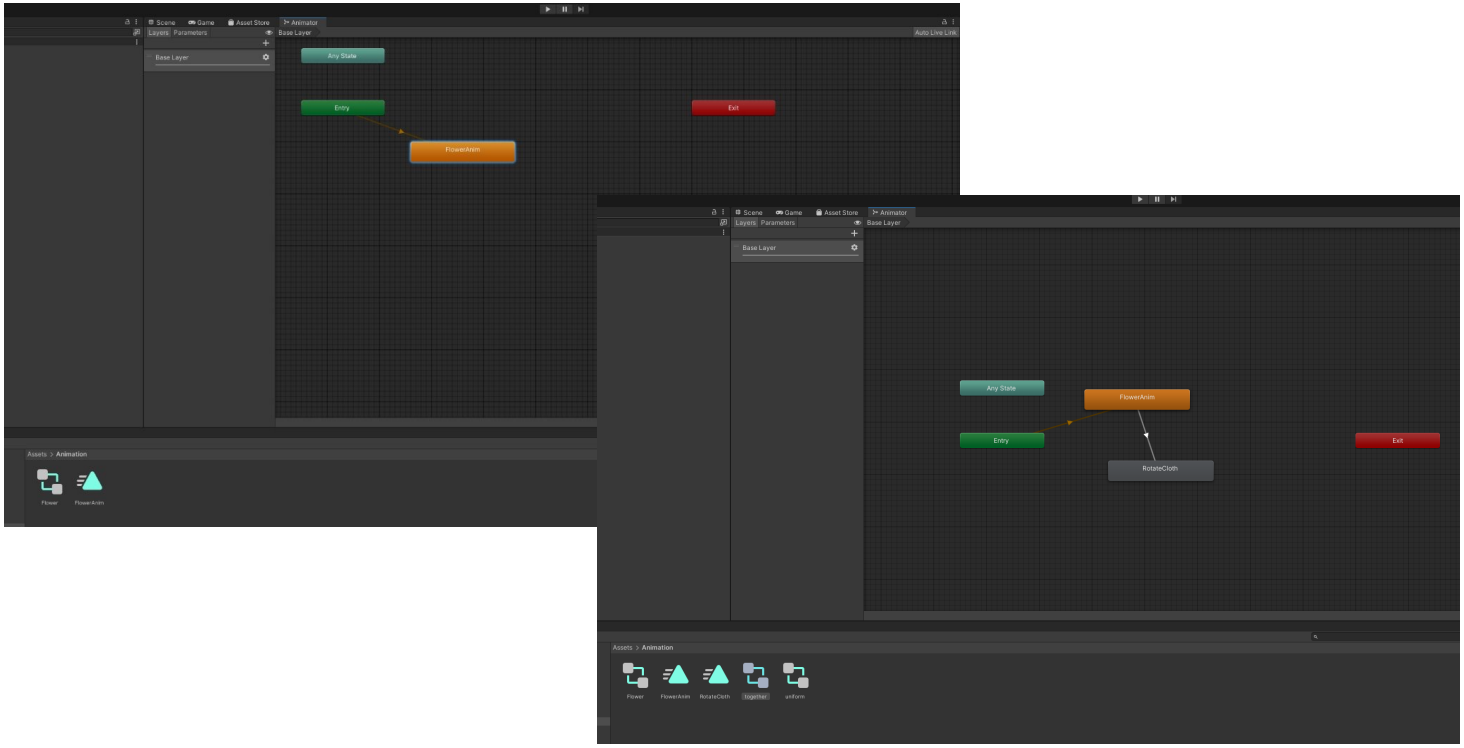
Animation Controllers

Adding Animation Clips

- Double-click the **Animator Controller** to open the **Animator Window.**
- Drag **animation clips** from the **Project window** into the **Animator Window** to add them to the controller.
- Each **animation clip** added to the Animator Controller becomes a **state**.
- The first clip added will be the default state (entry point) indicated by an orange color.

Creating Transitions

- Right-click on an animation state and select **Make Transition** to draw a transition arrow to another state.

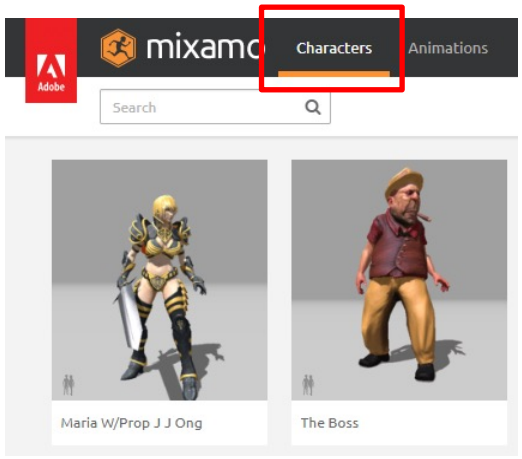


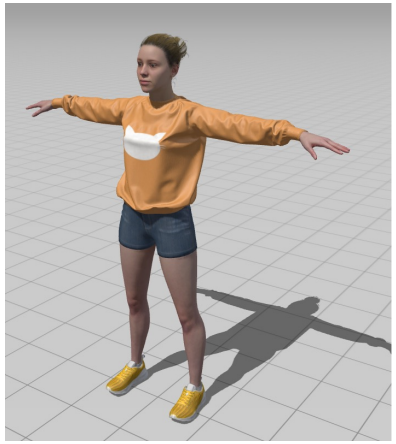
Creating a Sitting Idle Avatar with Mixamo and Unity:

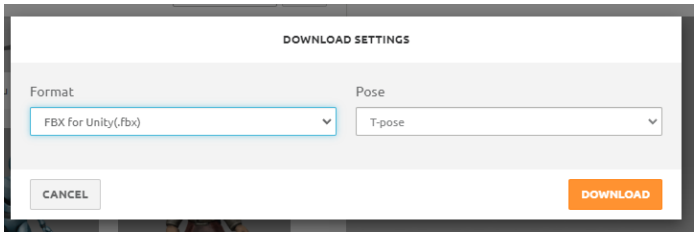
Mixamo:

- A web-based platform that offers free 3D character models and animations.
  https://www.mixamo.com/

- **Selecting and Downloading a Character from Mixamo:**
  - Create an account or log in if you already have one.
  - Browse and select a character model from the Mixamo library.
  - Click **Download** and choose **FBX for Unity** format, with T-Pose.



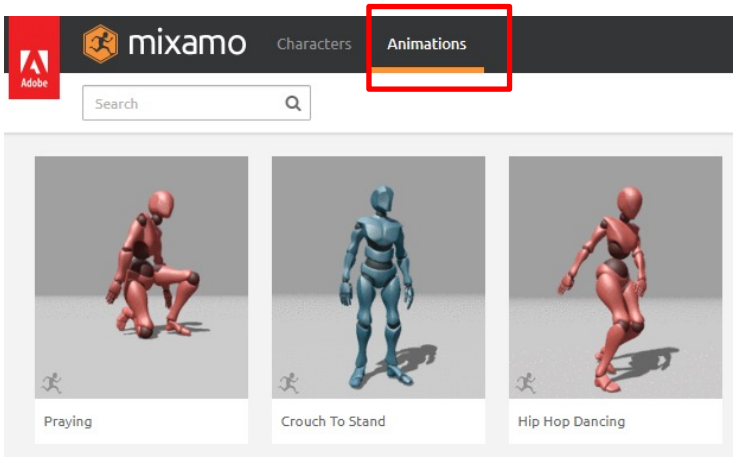


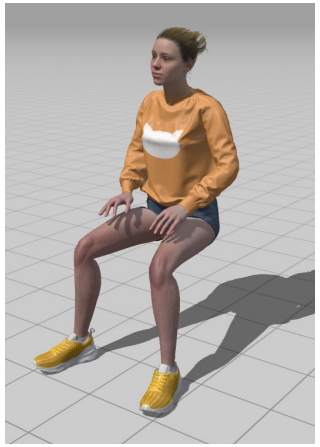


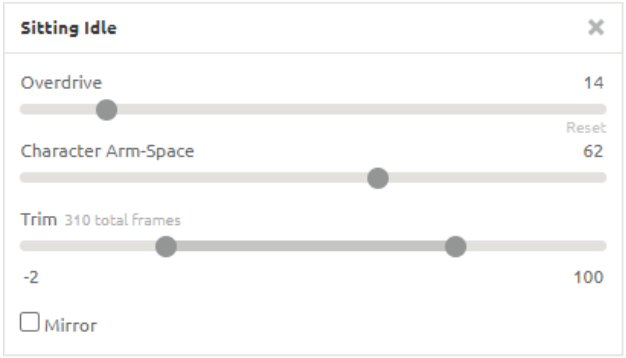
Creating a Sitting Idle Avatar with Mixamo and Unity:

➢ Applying the Sitting Idle Animation:

- In Mixamo, search for a "Sitting Idle" animation.
- Customize settings like arm spacing, as needed.
- Increasing the overdrive value speeds up the animation, making movements more rapid and energetic.
- Use the trim feature to remove any unnecessary frames from the beginning or end of the animation.
- Click **Download**, choose **FBX for Unity** format, and ensure the animation is set to "Without Skin".



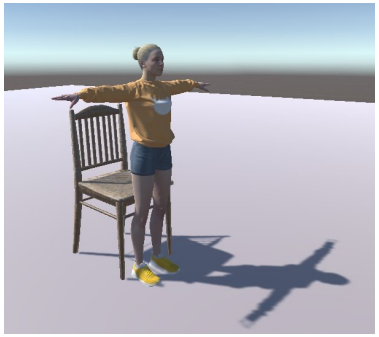


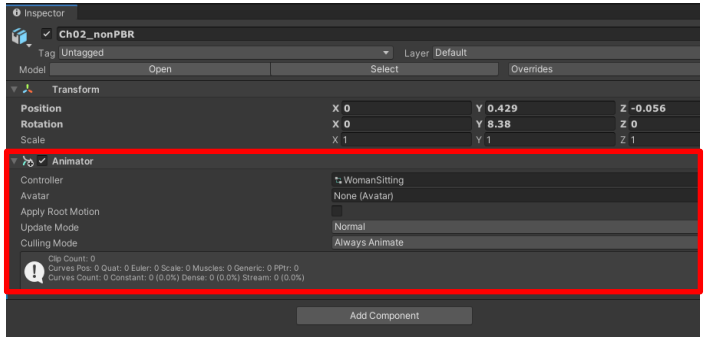


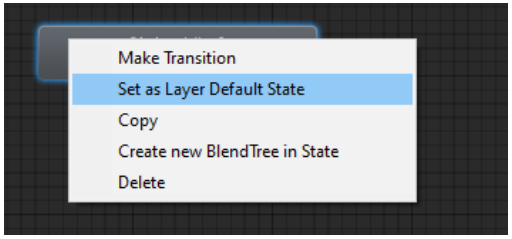
Creating a Sitting Idle Avatar with Mixamo and Unity:

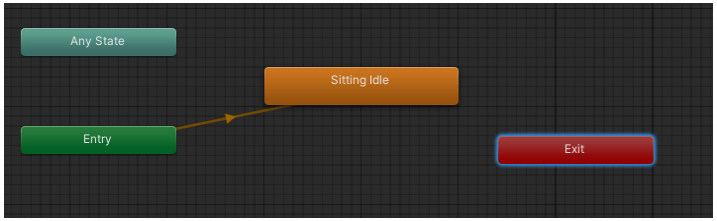
Setting Up the Character in Unity:

- Drag the character model into the Scene view.
- Create an Animator Controller by right-clicking in the Project window and selecting **Create > Animator Controller**.
- Add the Animator component to the character.
- Assign the Animator Controller to the character's Animator component.
- Double-click the Animator Controller to open the Animator window.
- Drag the sitting idle animation clip into the Animator window to create a new state.
- Right-click the new state and select **Set as Default State**.
- Import a chair model into Unity and place it in the Scene view.
- Adjust the position and rotation of the avatar so it sits correctly on the chair.
- Enter Play mode to see the avatar performing the sitting idle animation on the chair.





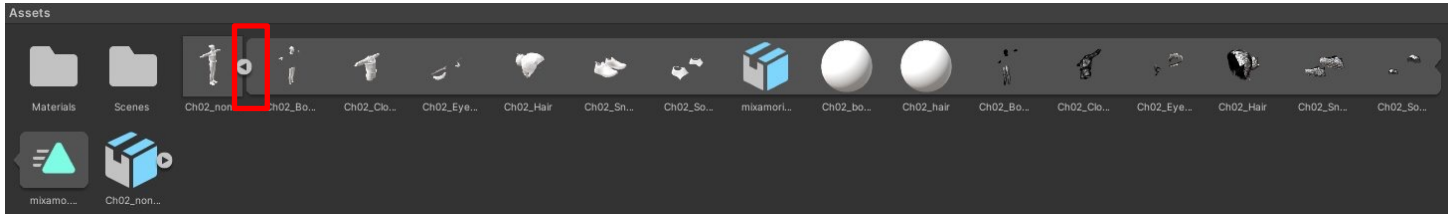


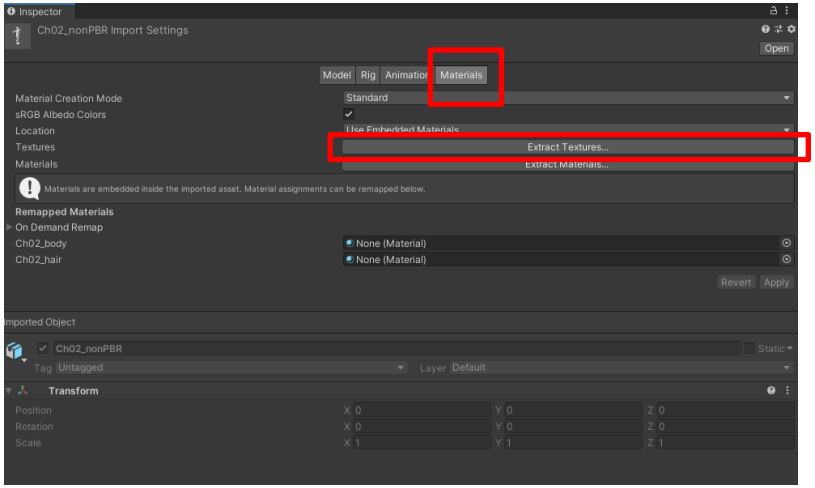


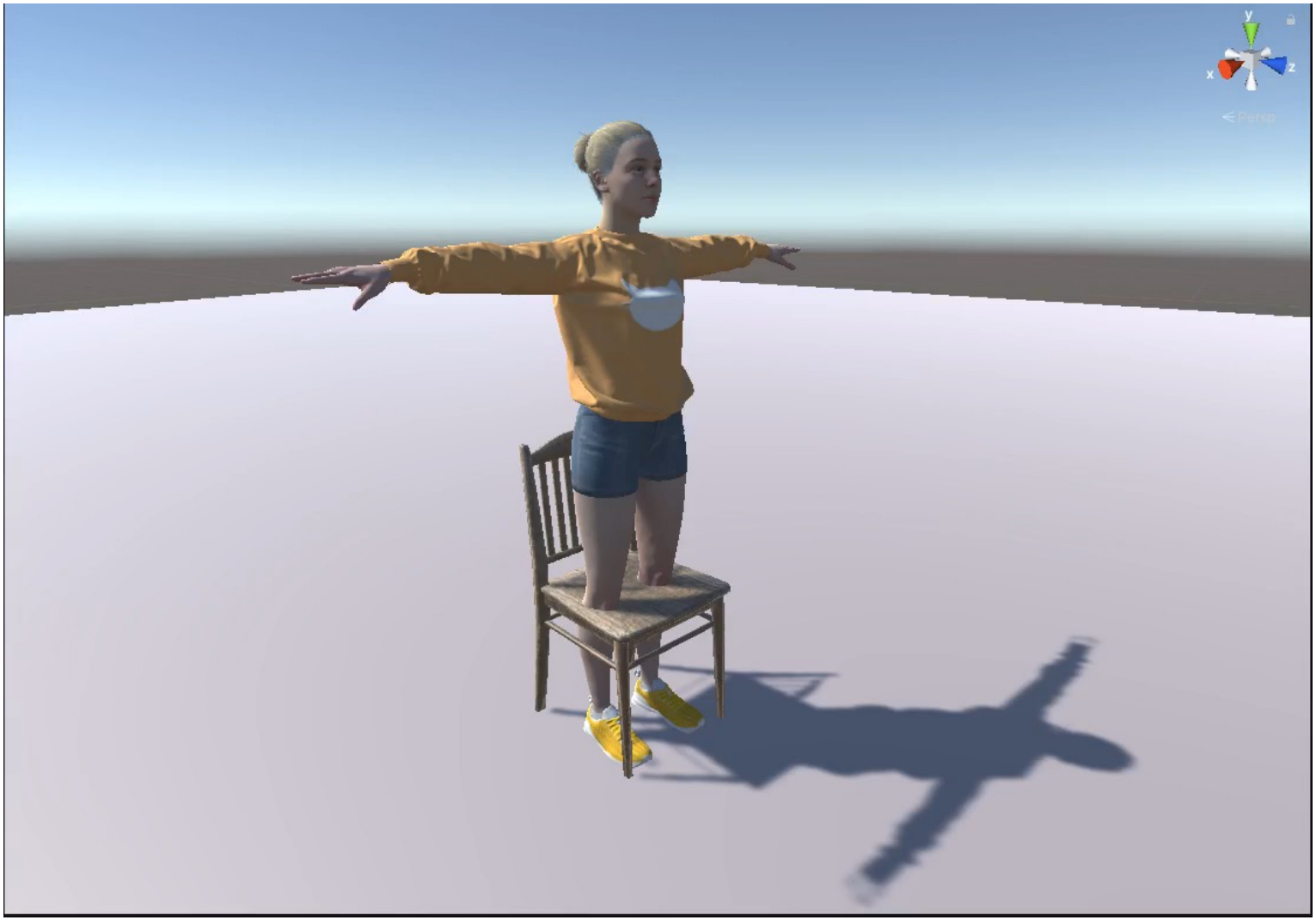
Correcting Model Textures and Colors:

If the model textures and colors are not displaying correctly after import. The reason is the Missing texture references or incorrect material settings.

1. Click on the model in the Project window to select it.
2. With the model selected, look at the Inspector window on the right side of the Unity interface.
3. In the Inspector, find the **Materials** section.
4. Click on the **Extract Textures** button.
5. Select a destination folder within your project to save the extracted textures.
6. Then, the model will be shown correctly.







Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- **Working with Lights**
- Setting the Skybox

Lighting

- Like the real world in Unity, we use **lights** to illuminate our scenes. Without **lights** nothing would be visible to our camera in our scenes. Lights are an essential part of every scene. Lights define the color and mood of your 3D environment.

- You'll likely work with more than one light in each scene. Making them work together requires a little practice but the results can be quite amazing.

We have two different types of lighting.

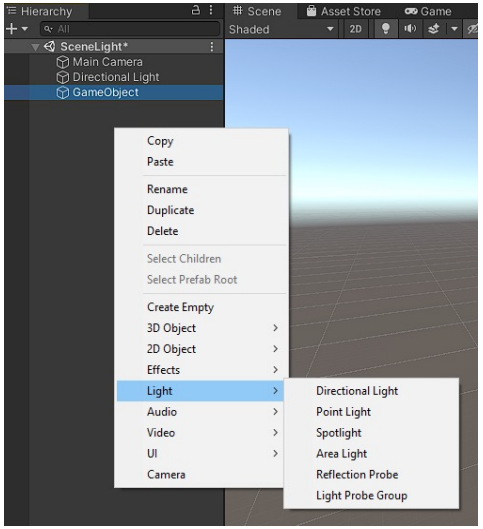
- Dynamic lighting
- Baked lighting

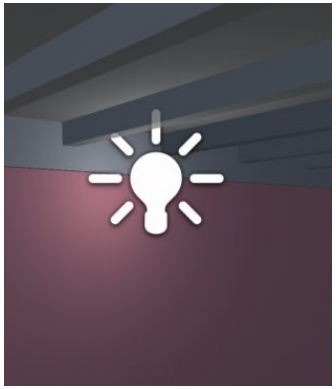
- **Dynamic lighting** is calculated in real time while our game is running. This can be achieved simply by adding lights to our scenes.

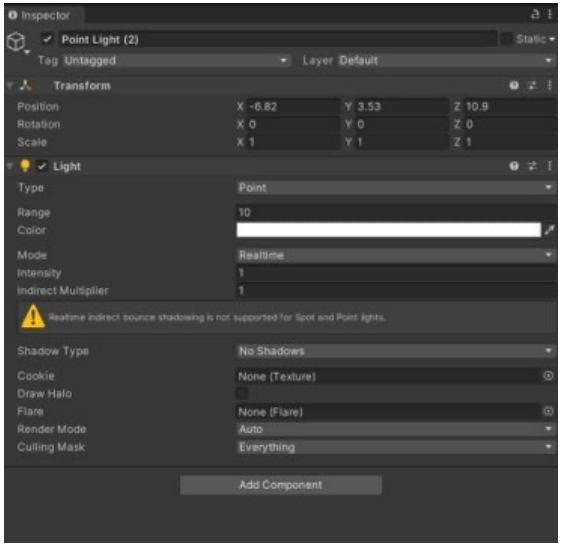
- **Baked lighting** is calculated offline and saved to a texture. These lighting texture maps then would be applied to the baked objects for improved performance.

How to add lights to our scene?

- **Lights** can be added to your scene from the right click on **hierarchy window -> Light** menu. You will choose the light format that you want from the sub-menu that appears. Once a light has been added, you can manipulate it like any other Game Object.



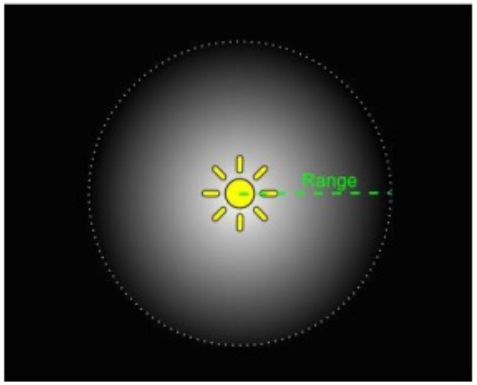




- In another way, you can add a Light Component to any selected Game Object by using **Add Component->Rendering->Light** and then select the type from the **Type** slot.
- By simply changing the **Color** of a light, you can give a whole different mood to the scene.

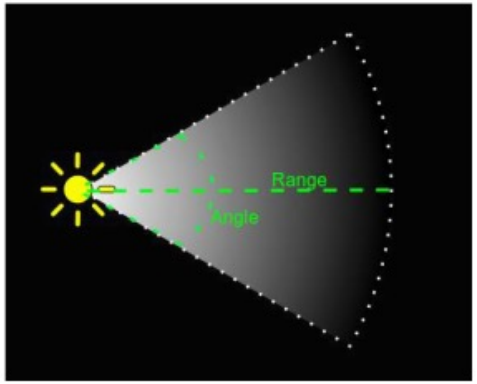
Types of lights

- **A point light** behaves like a bare light bulb. It illuminates objects in the scene based on its position in the scene. Rotation has no influence on the illumination, and it sends light out in all directions equally. The intensity diminishes with distance from the light, reaching zero at a specified range.



To check: Range, Color, Intensity.

- **A spotlight** behaves like a flashlight or a headlight on a car, points to a direction based on The transform's rotation and illuminate all the objects within the cone. However, the spotlight is constrained to an angle, resulting a cone-shaped region of illumination. Light also diminishes at the edges of the spotlight's cone. It responds to both rotation and position.

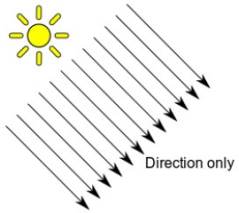


To check: Range, Color, Spot Angle, Intensity.

Types of lights

- **Directional Light:** it behaves like the sun and can be thought of as distant light sources which exist infinitely far away. It effects all the objects in the scene based on its rotation. Its position in the scene does not influence anything. Hence the distance of the light from the target object is not defined and the light does not diminish.

To check: Direction, Intensity.



- **Area Light:** it only works when baking a light map. It shines in all direction to one side of a rectangular plane.

An area light is a rectangular that produces light with a soft shadowing.

Part 1 : Unity for creating virtual environments (essentials)

- Introduction to Unity as a Game Engine
- Getting Started with Unity
- Creating and Managing Basic Game Elements
- Multimedia Integration and UI in Unity
- Working with the ProBuilder Tool in Unity
- Working with Lights
- **Setting the Skybox**
  - **Setting the Skybox to a Panoramic Image**
  - **Setting a Procedural Skybox**

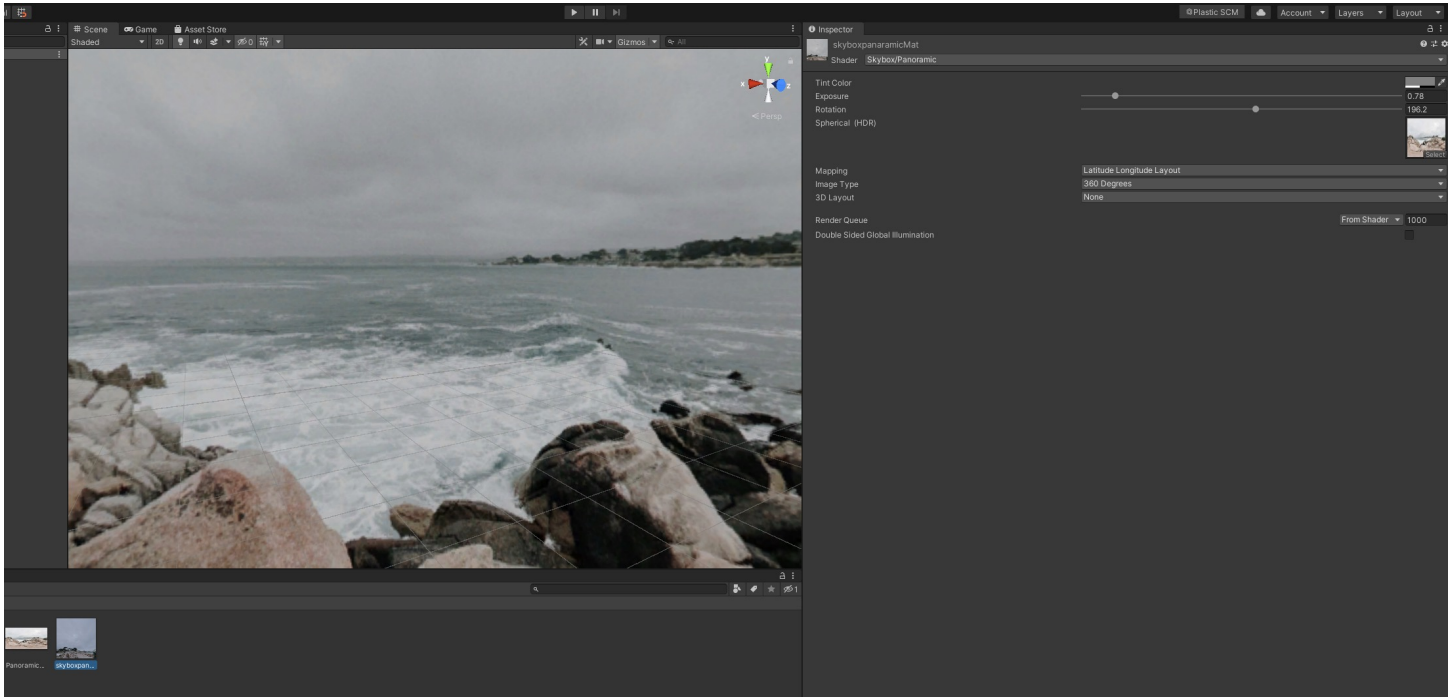
Setting the Skybox to a Panoramic Image

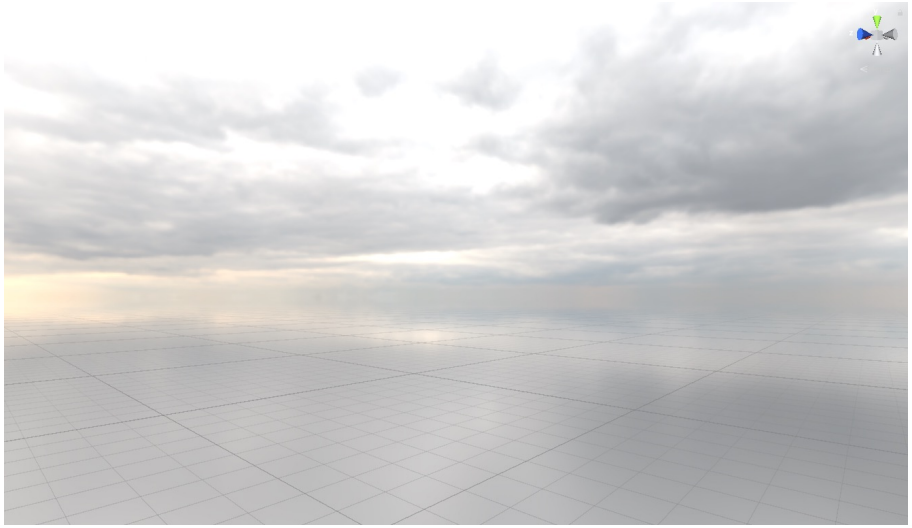
- A Skybox is a background that surrounds your entire scene, providing a sense of environment.
- Used to create realistic or artistic skies, horizons, and backgrounds.

Setting the Skybox to a Panoramic Image:

1. Import your panoramic image into Unity by dragging it into the Project window.
2. Ensure your image is in a supported format, such as JPG or PNG.
3. Right-click in the Project window and select **Create > Material**.
4. In the Inspector, change the Shader to **Skybox > Panoramic**.
5. Drag your panoramic image into the **Spherical (HDR)** slot of the material.
6. Configure additional settings like Exposure, Rotation, and Mapping if needed.
7. Go to **Window > Rendering > Lighting**.
8. In the Lighting window, under the **Environment** tab, assign your new Skybox material to the **Skybox Material** slot.

Setting the Skybox to a Panoramic Image



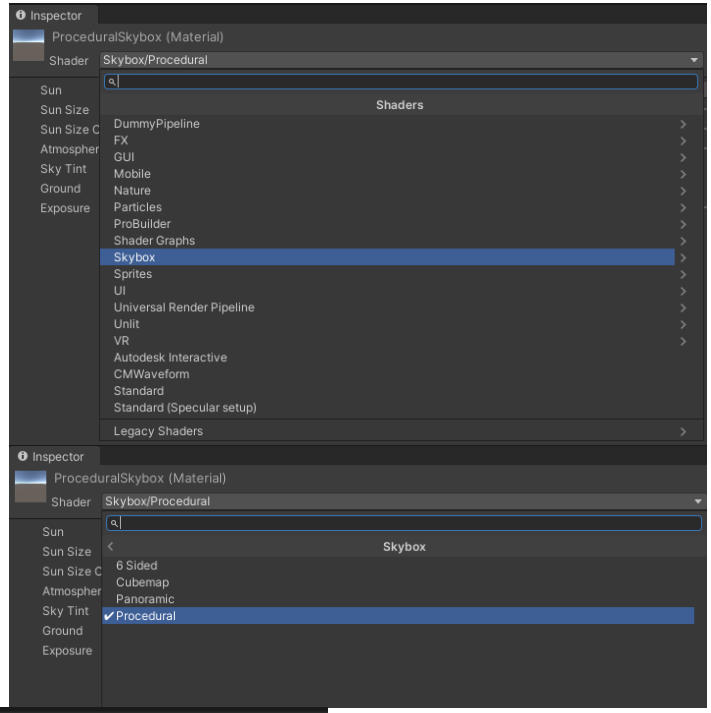


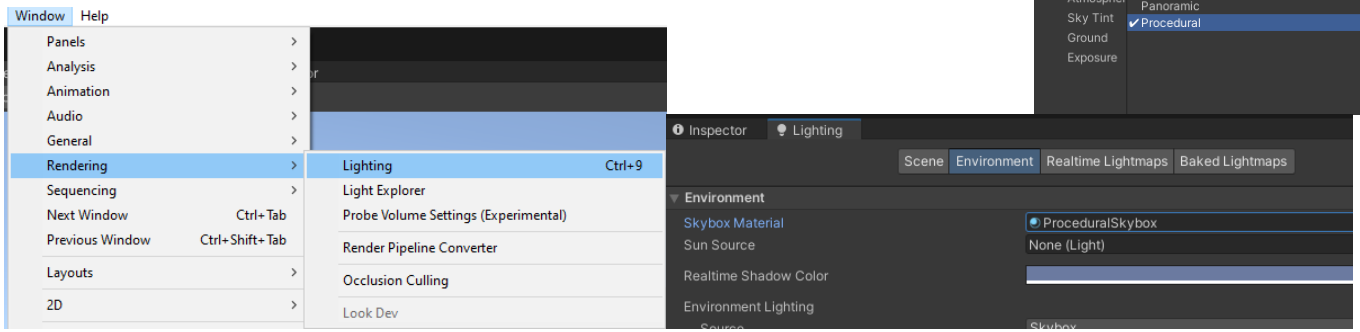
Setting a Procedural Skybox:

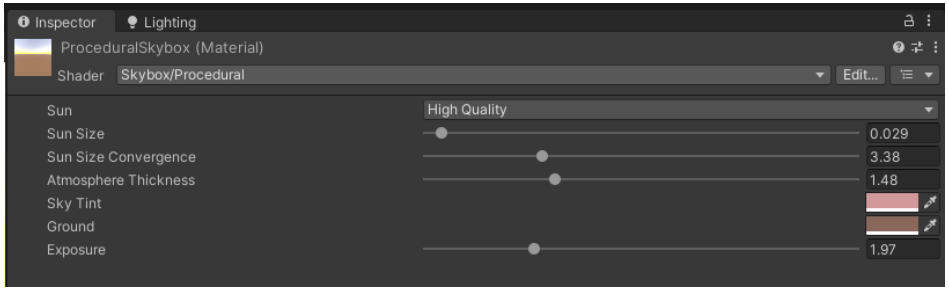
1. A Procedural Skybox is a dynamic sky that can change over time, ideal for simulating day/night cycles.
2. It enhances realism by allowing for environmental changes based on time or events.

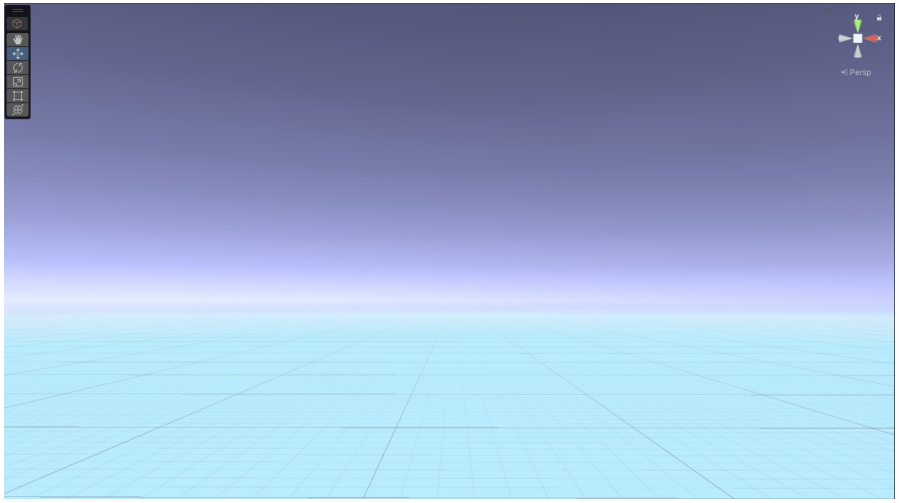
Steps to Create:

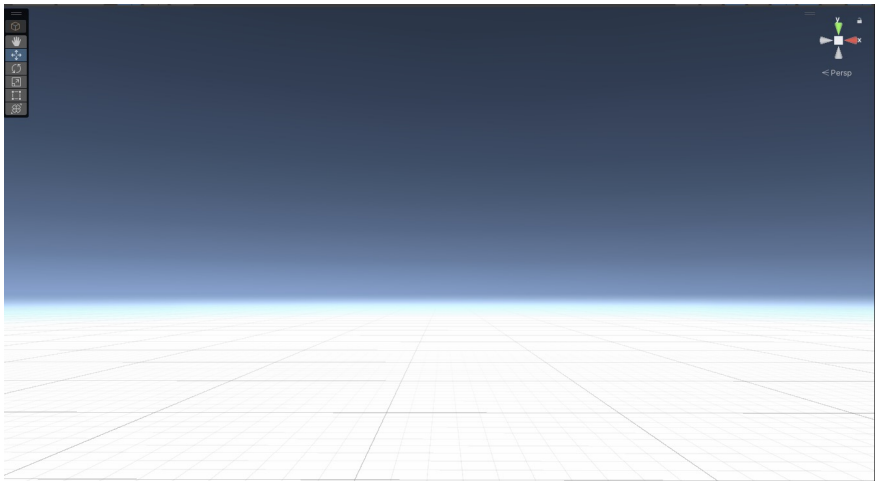
1. Right-click in the Project window and select **Create > Material**.
2. Change the Shader to **Skybox > Procedural.**
3. Name your new material, e.g., "ProceduralSkybox".
4. Control the size of the sun in the sky.
5. Adjust the density of the atmosphere.
6. Change the color of the sky.
7. Set the color of the ground as seen in reflections.
8. Control the overall brightness.
9. Go to **Window > Rendering > Lighting**.
10. In the Lighting window, under the **Environment** tab, assign your ProceduralSkybox material to the Skybox Material slot.





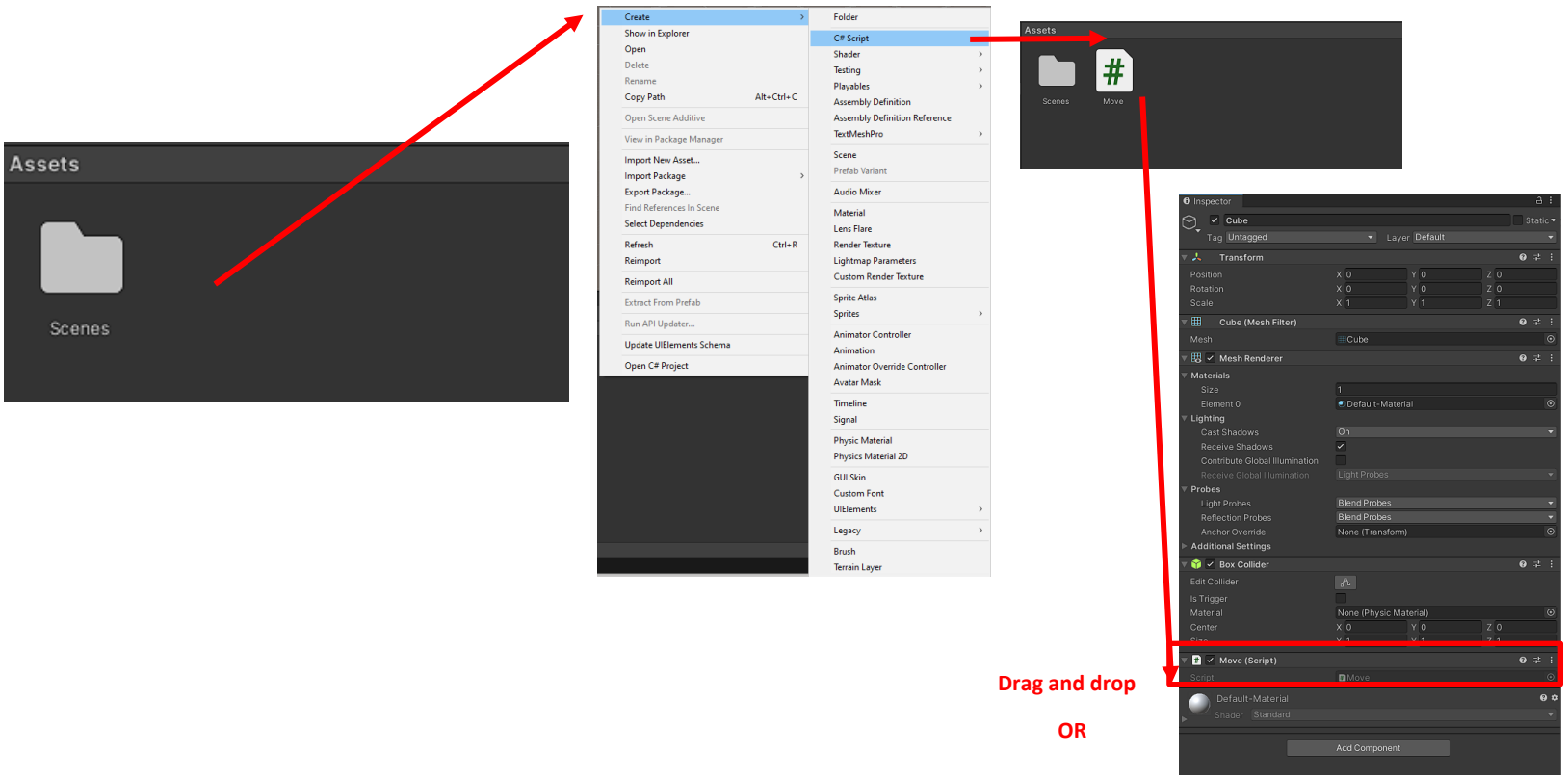






How to create a script and add it to a Game object?

1. Right click on the **Assets window -> Create -> C# Script**.
2. Double click on the file created to open it in the **Visual Studio**.
3. You can attach this script by drag and drop it on a Game Object or through the **Add Component** button.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Shirin Hajahmadi

Shirin.hajahmadi2@unibo.it