CODE NINJAS®

Bronze Belt Ninja Guide

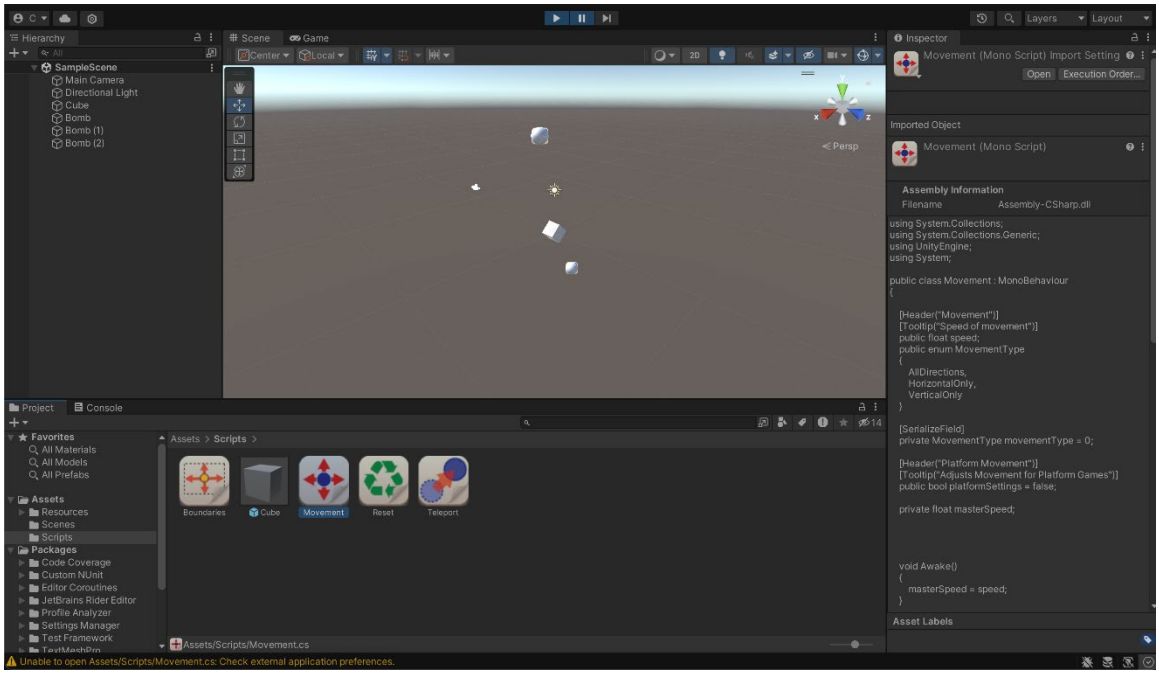
Activity 01: Dropping Bombs

Welcome to Bronze Belt!

Welcome to Bronze Belt! You've reached a significant milestone in your journey. It's a great accomplishment, and you should be proud of yourself. Everything that you've learned up to this point has been in preparation for this.

From now on, you'll be working with the same tools that professionals use to make games. This might seem a bit overwhelming at first. There are so many different options and ways to do things that it might make your head spin at first. Don't forget what you learned before in the previous belts. At the heart of every lesson are the same core programming concepts. You're still using variables, conditionals, and functions. The difference with Unity is that you have so many more opportunities than you had before.

But before we begin, what is Unity?



Working in Unity

Unity is the same tool used by professionals to make games such as Pokémon Go, Fall Guys, and Among Us. With Unity, you can bring various elements together to build a complete game that can be played on different devices and platforms.

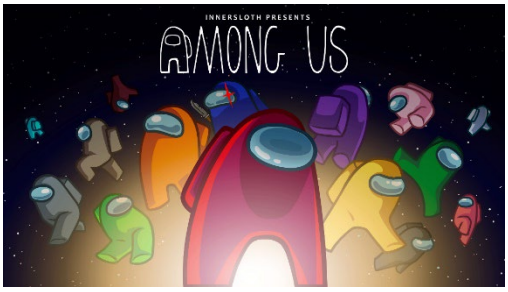


Image Sources (Left to Right): 1) Among Us: Epic Games, 2) Fall Guys: Fall Guys | Download & Play Fall Guys on PC for Free – Epic Games Store, 3) Pokemon GO: Pokémon GO (pokemongolive.com)

What do I need to know?

To get you started, we will concentrate on the Unity interface, how to find things and what to do with them. For the first few activities, you will use pre-built scripts to make your games. This approach will ease you into Unity's environment until you are familiar enough with Unity to start working with the programming language, "C Sharp" (C#).

Unity utilizes C# because it seamlessly aligns with the components of Unity game objects. Furthermore, as you'll soon discover, you can attach multiple C# scripts to a single object, facilitating the writing of code in manageable, bite-sized portions.

A Brief Word About 2D and 3D

Games are typically developed in either a 2-dimensional (2D) or 3-dimensional (3D) environment.

In a 2D environment, your focus lies in the x and y axes, governing an object's horizontal and vertical movement. The x-axis represents left and right movement, while the y-axis represents up and down movement. However, it's worth noting that the z-axis can also be employed in 2D to position or layer objects in depth, albeit without the perception of 3D space.

On the other hand, 3D introduces an additional dimension, the z-axis, which allows for forward and backward movement. Unity provides the flexibility to create games in either a 2D or 3D environment, even though Unity's core architecture remains 3D. You can initiate a Unity project in 3D and utilize a 2D (orthographic) camera and 2D colliders to craft a 2D game using 3D objects, as we'll demonstrate later in this section.

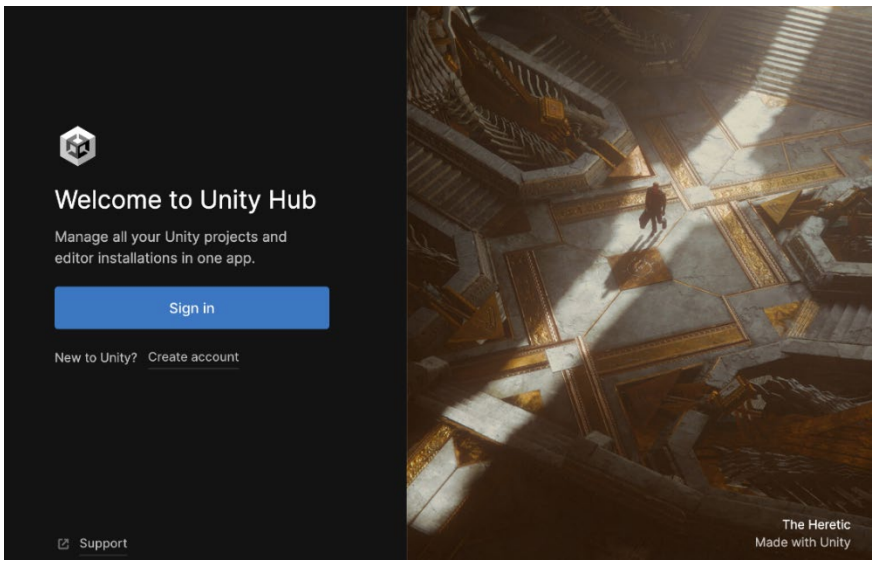
While all the activities in this book start as 3D projects, the initial ones exclusively involve the x and y axes, helping you understand that objects in Unity can take on various dimensions as needed!

Getting Started with Unity

1 To start, locate **Unity Hub** on your desktop. The icon should resemble something like the image below. Move your cursor to Unity Hub and click it twice to open the app.

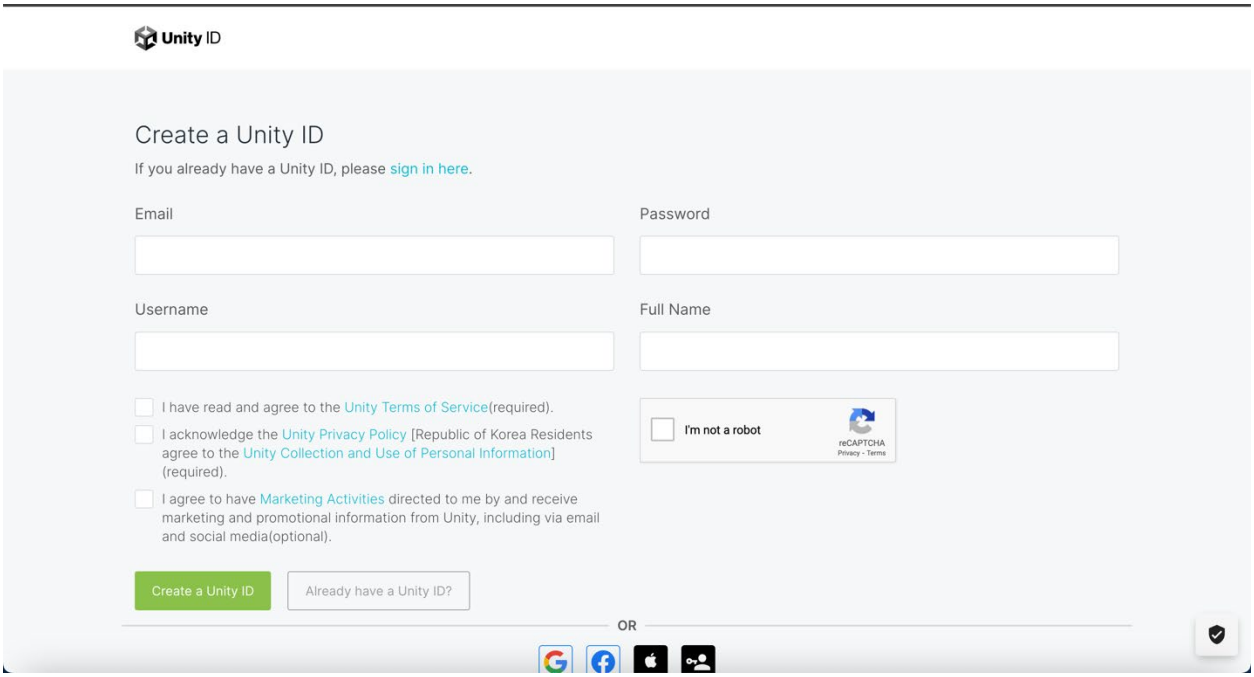
If you can't locate Unity Hub on your desktop, let a Code Sensei know and they will assist you!

2 Once you click Unity Hub it will appear, and you will be greeted by a screen that looks like this. Please note, the screen appearance may differ as Unity has a Black and White appearance option.



3 If you already have a Unity account then you can select the "**Sign in**" button otherwise click on "**Create account**". Remember, if you are under the age of 13, you will need to get permission to create a Unity account. If you're having trouble, have your Code Sensei help you.

When you click on the create account button, you will see a screen like the one below:

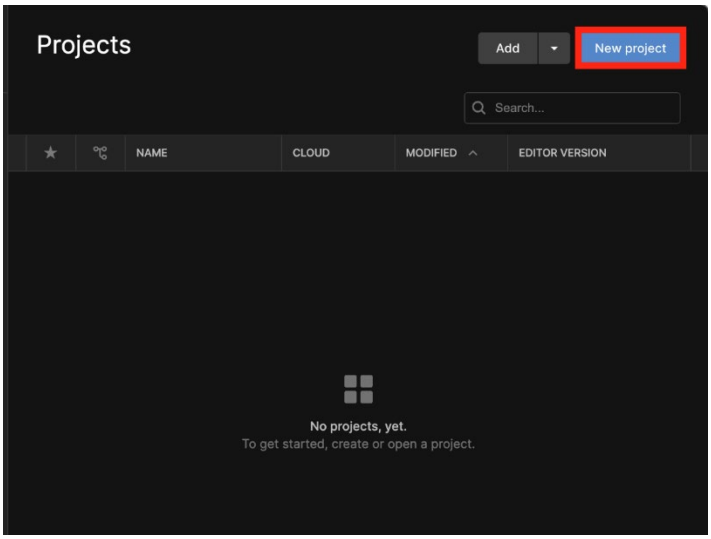


You can fill in your details or have a Code Sensei help you do this. You can also sign on using a single sign-on provider, such as Google or Apple. Once you finish your details, click **Create a Unity ID**. Then return to Unity Hub.

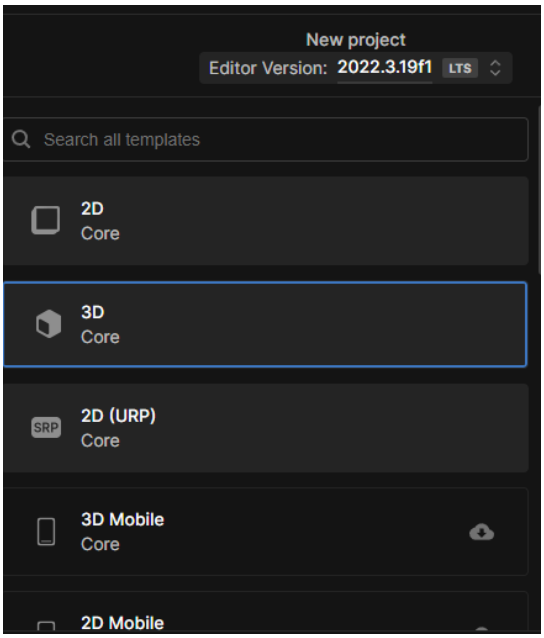
Once your account is created, sign into Unity Hub with it.

4 Once you are signed in, you will see the Projects window.

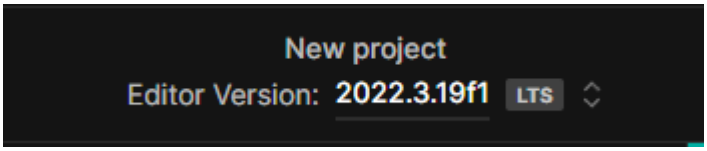
When you first start, you will not have any projects to open. That's okay. Make a new one by clicking the blue "**New project**" button.



5 On the left side of the screen are options for what type of project you will be creating. You will most often be choosing 2D or 3D. Go ahead and leave 3D selected.



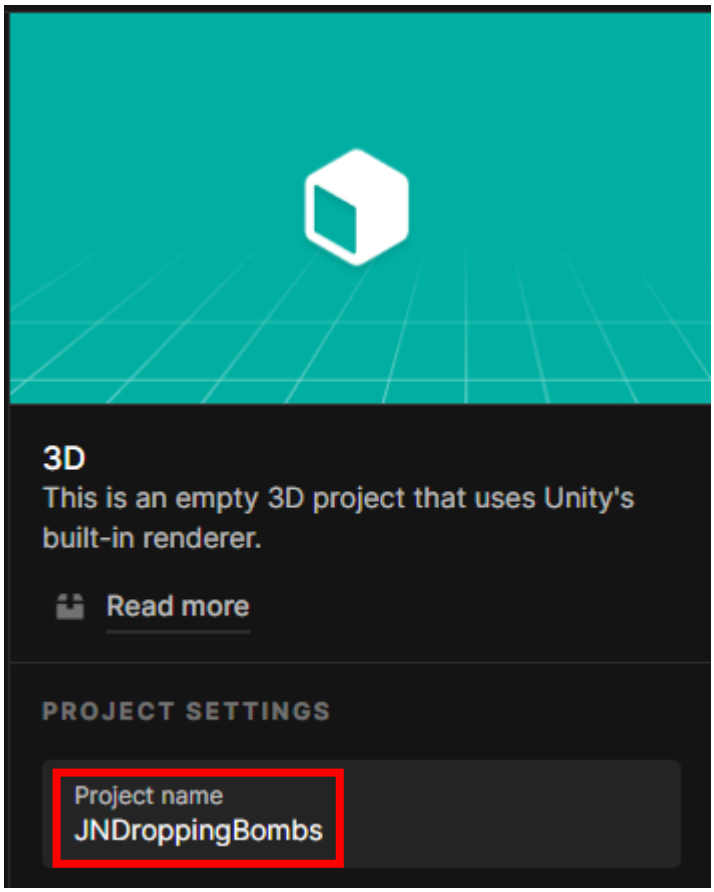
6 Ensure that the **Editor Version** at the top of the screen states "**2022.3 LTS**"



Using this version will ensure everything looks the same as you follow along in this guide.

7 You will also need to select a name for your project. This is very important: you *cannot* change the name of a project once it is created. Make sure that it describes the project well.

Your first project is **Dropping Bombs**, so select a name that reflects that. For example, if your name is John Ninja, why not use your initials and name your file **JNDroppingBombs**.



8 Finally, you need to choose where to store your Unity projects. It is best to create a new folder with a name like **John Ninja's Unity** or **JNUnity** to keep everything in one place.

However, your Code Ninjas location may handle Unity project storage differently. **Ask your Code Sensei** where to save your Unity projects before proceeding to the next step!

9 All that's left to do is click on **CREATE** and let Unity set up the files for your first project! Once that's done, move to the next page to create your first game in Unity!

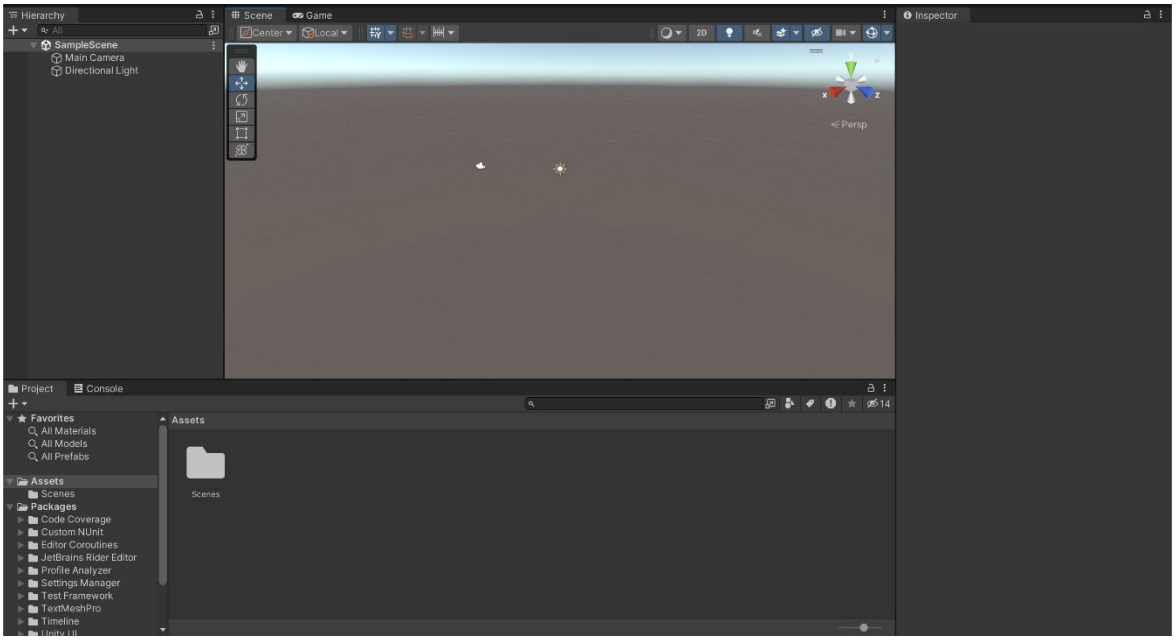
Pro Tip:

Projects in Unity are saved in the location that you've chosen as a folder with the name of your project. If you ever need to duplicate a project, just make a copy of the folder and all its contents and paste it and give it a new name. To open the new folder in Unity, click on Add and select the new folder.

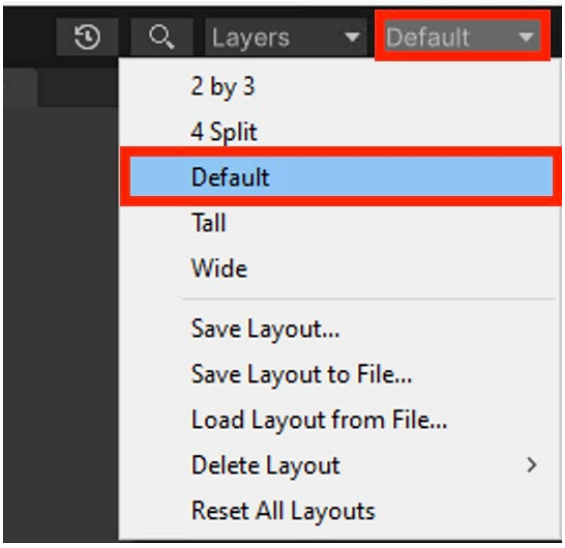
Activity 1: Dropping Bombs

This first project in Unity is meant to show you the possibilities achievable even without elaborate graphics, animations, or complex scripts. If you haven't done so already, open Unity and create a **new** 3D project, and give it a name like **MyInitalsDroppingBombs**.

1 After you have created your new project, Unity will open and show you something like the image below. Don't worry if it doesn't look *exactly* like this, we'll take care of that in the next step.

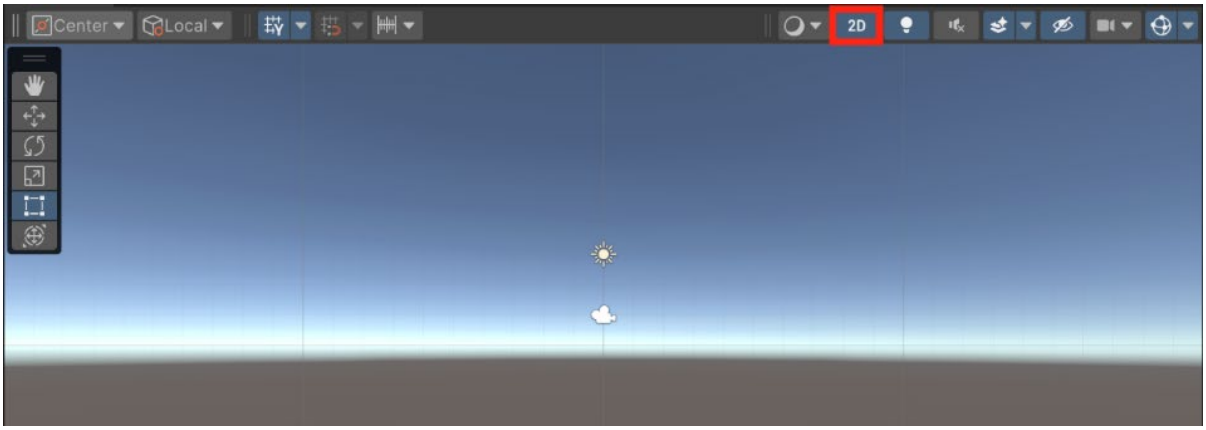


2 Unity gives you the option to rearrange the layout based on what works best for you. But for now, let's make sure you are using the default layout to avoid confusion. Click the **Layout** button located in the upper right corner. Select **Default** and your window should look just like the above image.

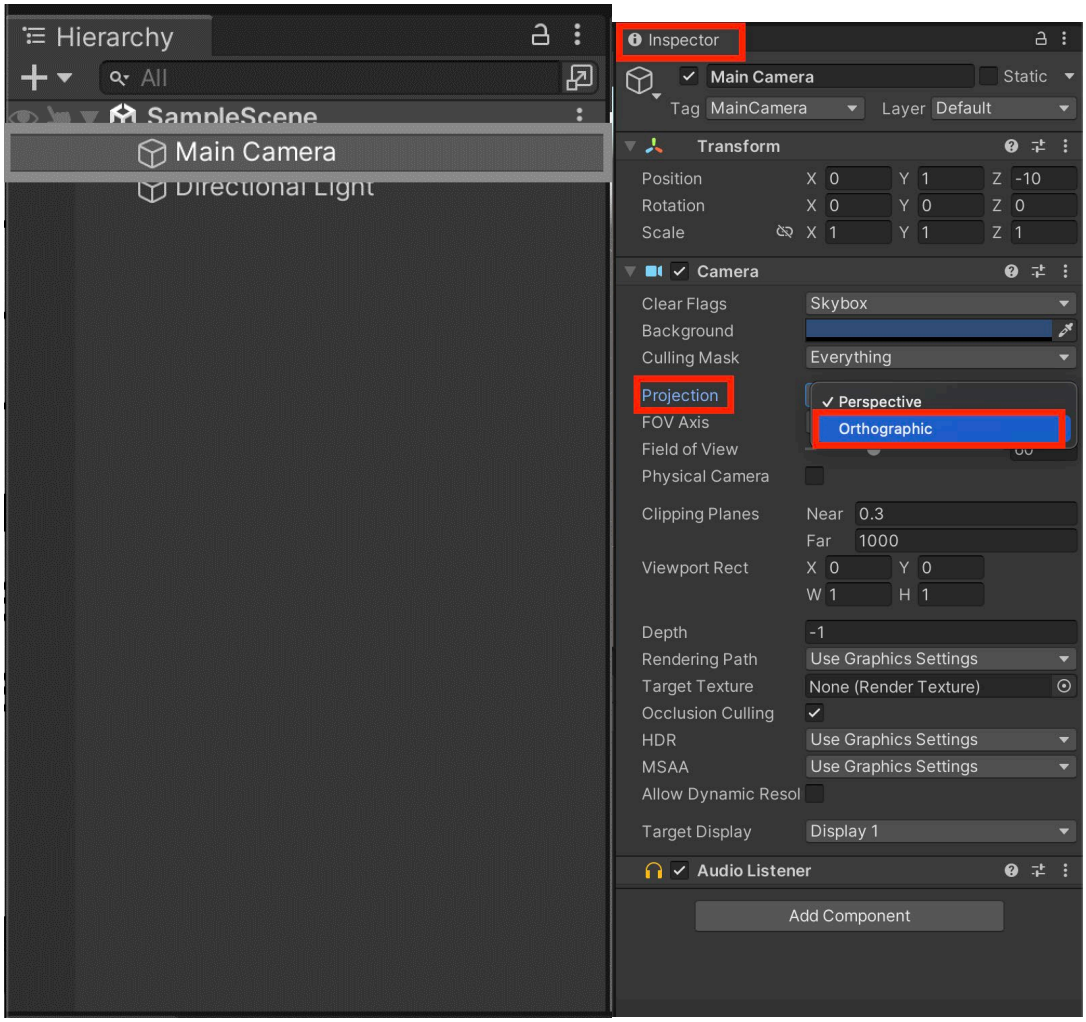


If you decide to rearrange the layout, you can use this same menu to save it for the next time you need it by clicking the "Save Layout..." button.

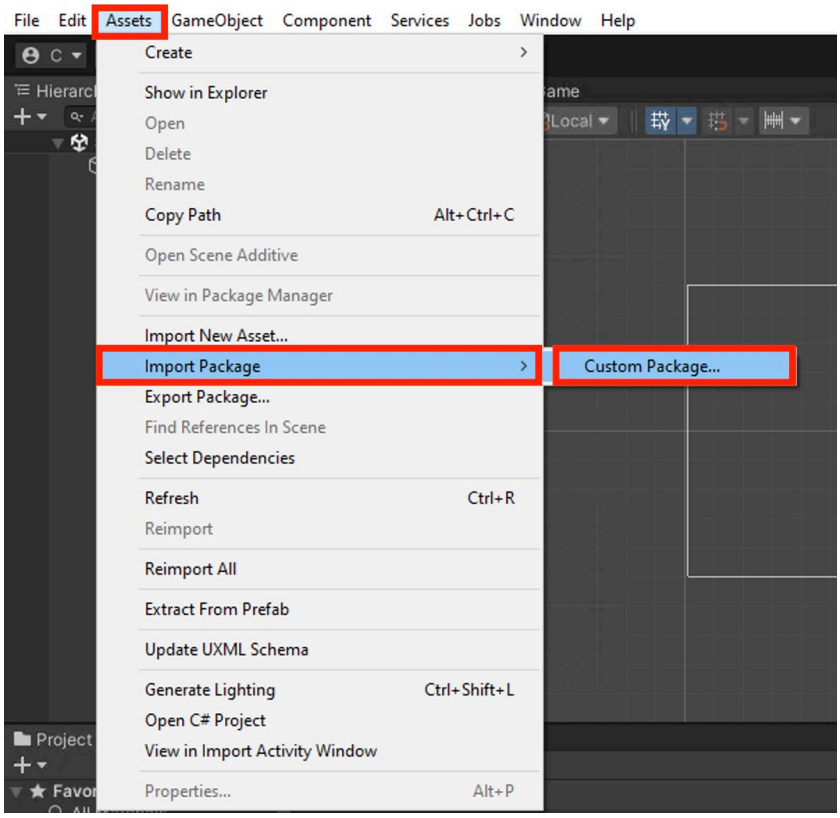
3 Even though this is a 3D Unity project, our game is only going to use movement in 2 dimensions (left to right, up and down). With that in mind, we can change the view so that it is strictly 2D. To do this, click the 2D button above the scene window as shown below.



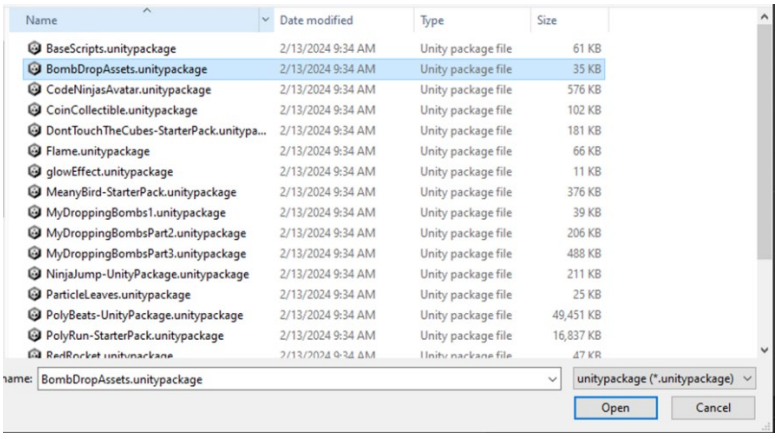
4 On the left side of the display is the **Hierarchy** panel. This shows you all of the **GameObjects** that are inside your scene. By default, Unity creates a Sample Scene and sets up a camera and light for you. Select the **Main Camera**. To the right of the scene, there is a panel called the **Inspector**. This panel contains all of the components associated with the selected **GameObject** in the **Hierarchy**. In this case, we want to change the **Projection** property of the camera from **Perspective** to **Orthographic** as shown below. We're doing this so that when we play the game, the objects will maintain their proportions and not get distorted by a 3D camera.



5 To make things easier, Code Ninjas has created some special scripts to use in this game. To load them, click **Assets** and select **Import Package** followed by **Custom Package**.

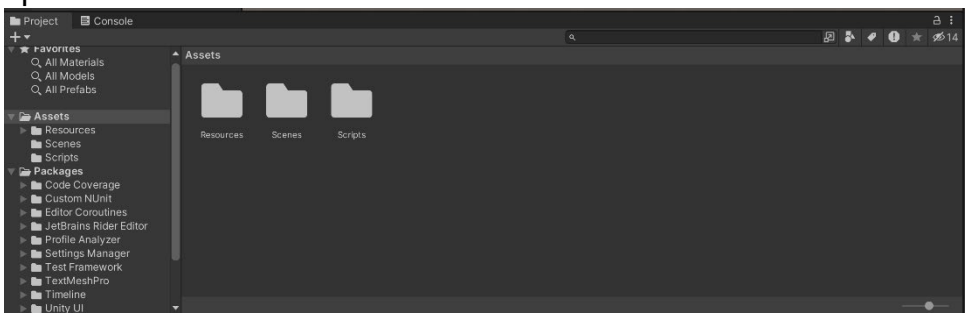


6 Navigate to the folder where your resource files are (ask your Code Sensei if you are uncertain) and select **Activity 01 - DroppingBombsAssets.unitypackage**.



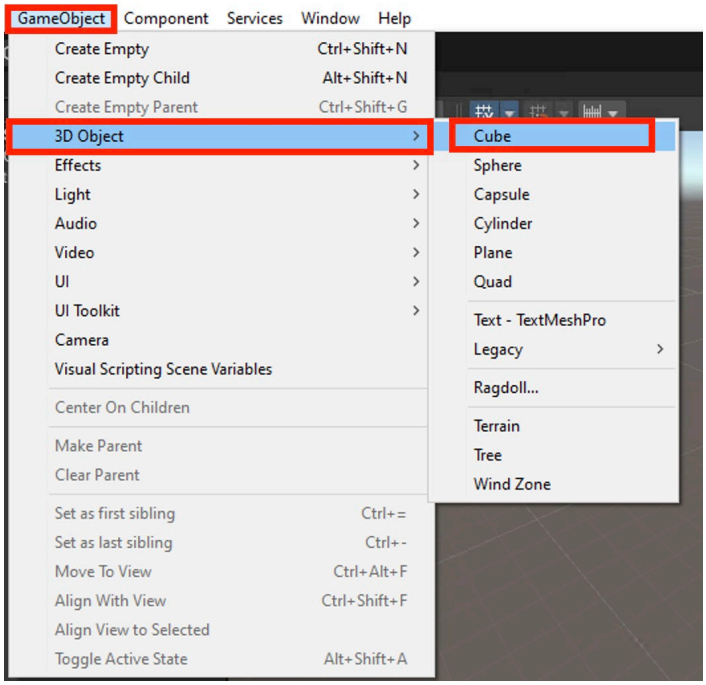
7 Unity will show you an import window with a list of the assets in the package. This gives you the option of loading just some of the assets instead of all of them. In this case, we want them all, so click **All** and then click **Import**.

8 The assets that you just loaded have been placed in the **Project** panel at the bottom of the Unity editor. The Project panel shows you everything that is a part of your project in folders, just as if you were browsing through files on your computer. In fact, these folders have been created as part of your project. Notice that you now have a new **Resources** folder and a **Scripts** folder that were imported with your package. You'll be using the scripts later.

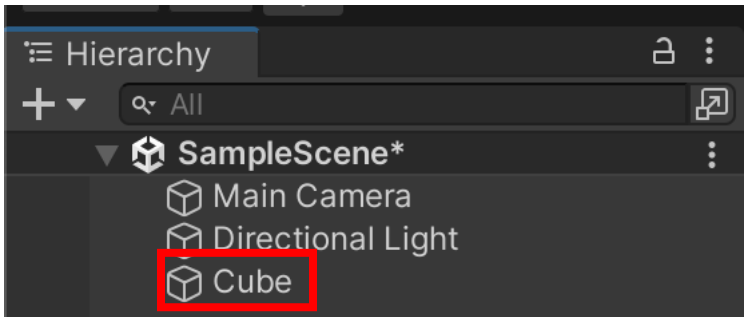


In Unity, the **Scene**, **Hierarchy**, **Inspector**, and **Project** panels are where you'll be doing most of your work, so feel free to take a few moments to look around each of these panels and become familiar with their functionalities.

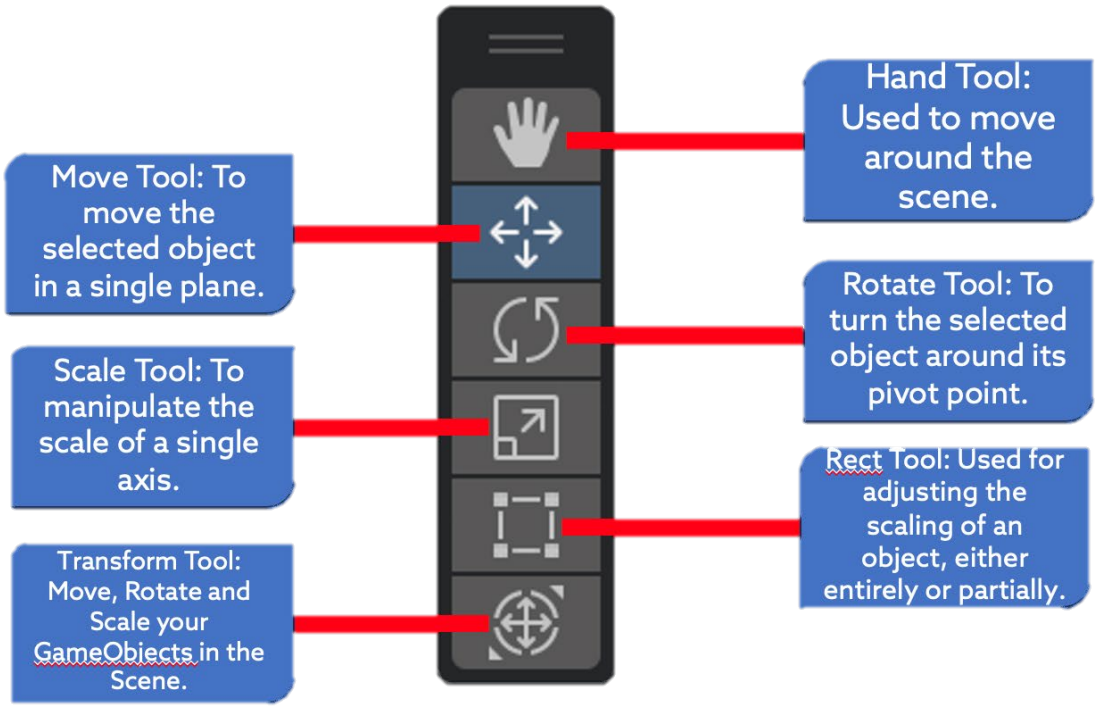
9 At the moment, the scene is a bit empty. Let's add a **GameObject** to the scene. Click the **GameObject** menu, select **3D Object** and **Cube** to add a cube to the scene.



10 You should now see a cube added in the **Hierarchy** panel. Select the cube (if it isn't selected already) to see the components for the cube in the **Inspector** panel.



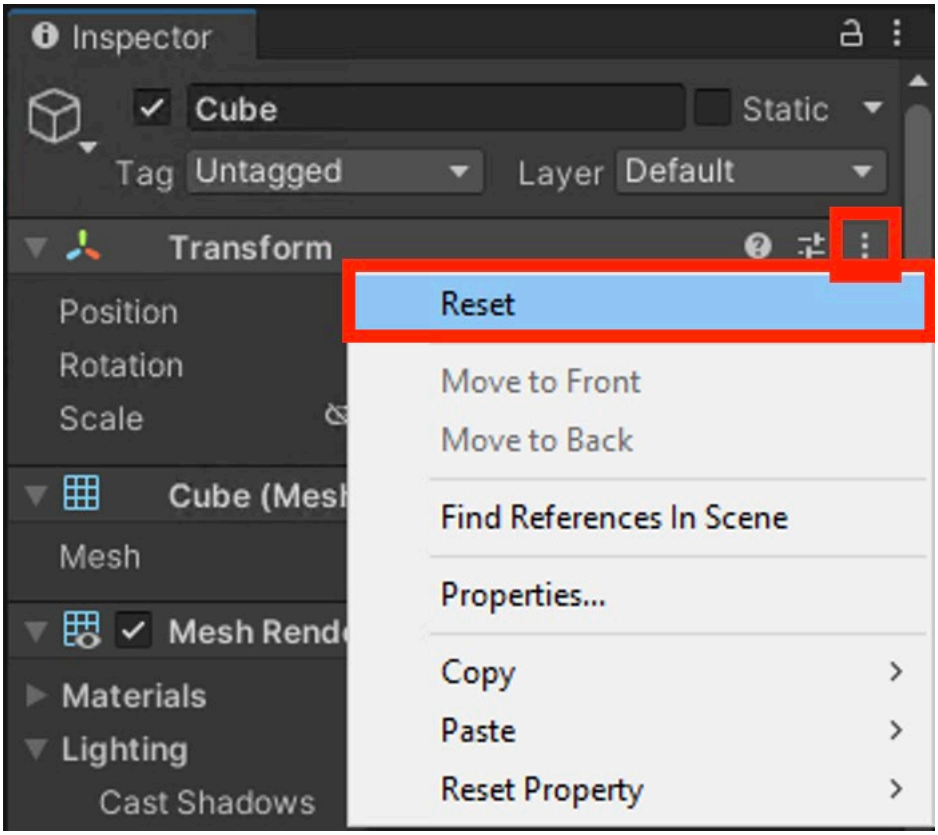
11 If you want to reposition, rotate, or make other changes to a **GameObject** within a scene, select the object in the scene by clicking on it and then using the tools above the scene to manipulate it. Go ahead and give it a try with the cube.



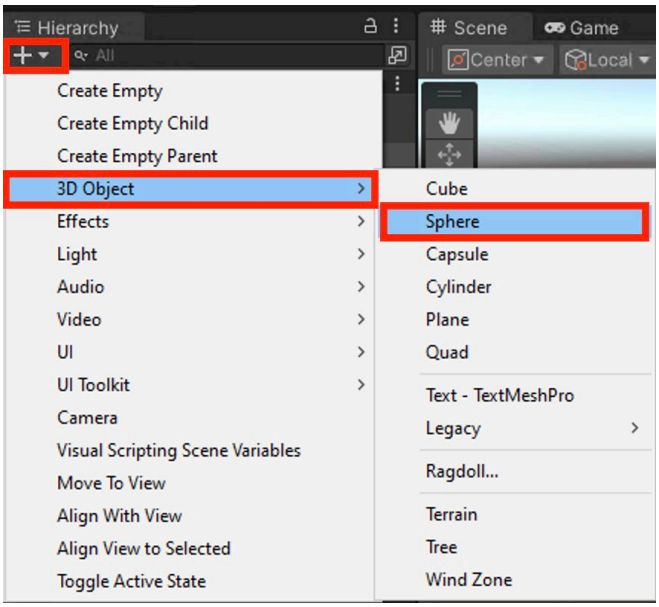
Pro Tip:

Regardless of the tool you're utilizing, navigating within the scene panel is easy using the mouse. Simply use the **mouse wheel** to zoom in or out of the scene. Furthermore, holding down the **right mouse button** and dragging allows you to move to the specific area of the scene you wish to interact with.

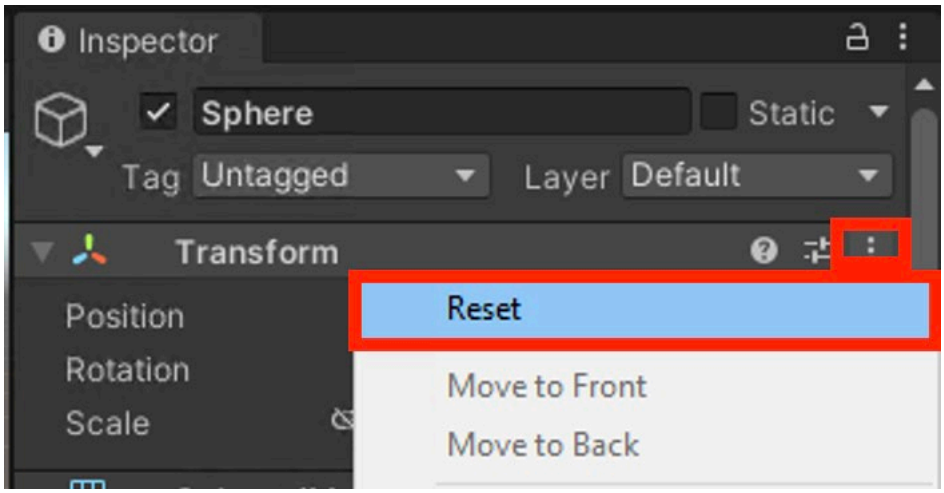
12 After you are done with the cube in the scene, it's best to return it to the middle of the scene at 0, 0, 0. With the cube selected, find the **Transform** component at the near top of the Inspector panel. This component shows the current position, rotation, and scale of the **GameObject**. A quick way to reset the transformations is to click on the three-dot menu in the upper right corner of the **Transform** component and then click on **Reset**. This should restore the cube's default settings.



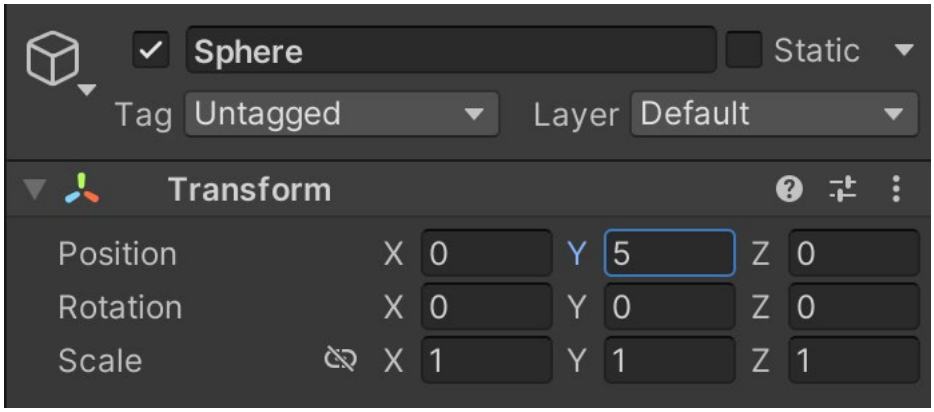
13 Now to add another **GameObject** to the scene. You can follow the same procedure as in the previous steps, but here's an alternative approach. In the **Hierarchy** panel, click the **Plus** button in the upper left corner and then click **3D Object**. Finally, select **Sphere** so we won't confuse it with the cube.



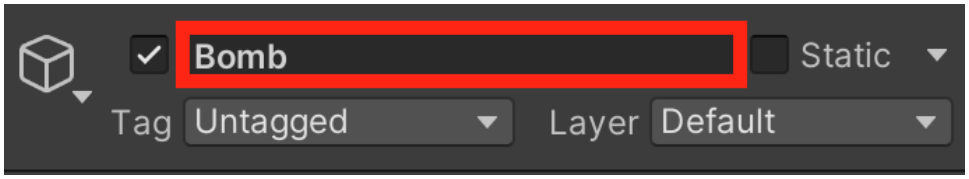
14 Let's make sure that the sphere starts out in the same spot as the cube. Use reset if necessary.



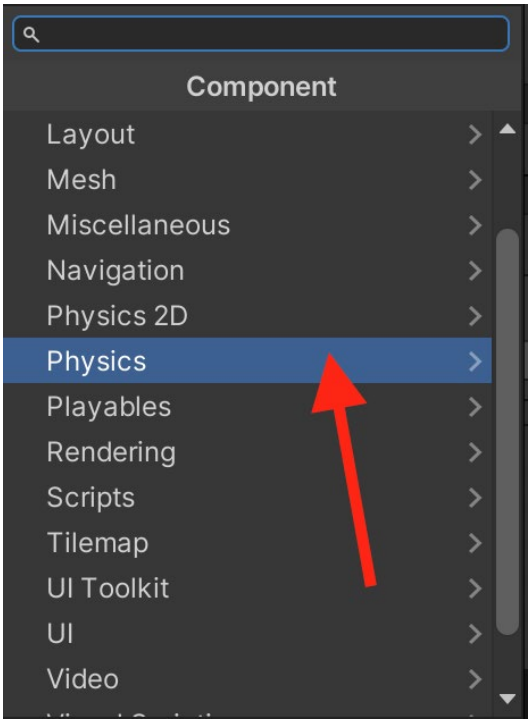
15 We want the sphere to start out above the cube. You can either use the move tool to move it in the scene or change the Y position of the sphere from 0 to 5 in the **Transform** component in the **Inspector** panel.



16 This game is called Dropping Bombs, not Dropping Spheres! Change the name of the sphere to ***Bomb*** by highlighting the name in the **Inspector** panel and typing in the new name.



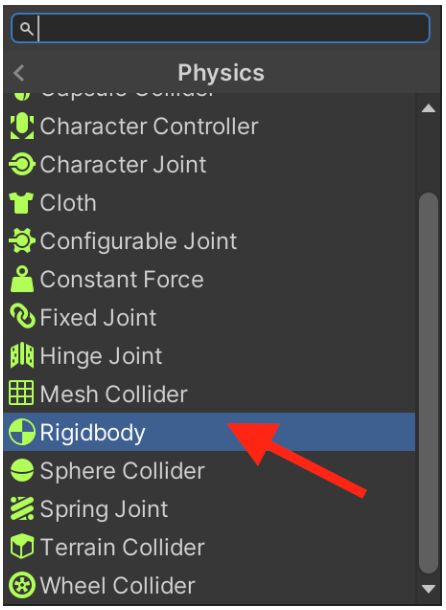
17 If you started the game now, not much would happen. Both game objects are just floating in space. If you want the bomb to drop, you'll need to have it respond to gravity. To do that, you'll need to add a new component to the object. With the bomb selected in the hierarchy, scroll to the bottom of the **Inspector** panel and click **Add Component**. This opens a menu of all sorts of new components and behaviors that can be added to the game objects. The component that you'll need is in **Physics**, so select that section.



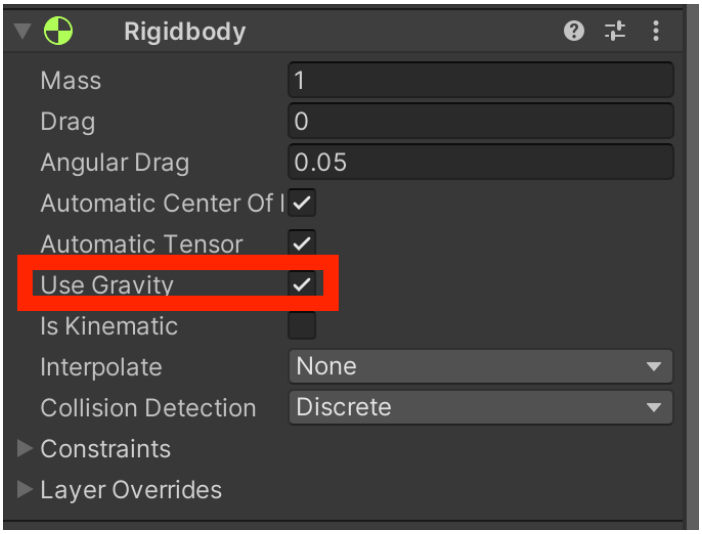
Pro Tip:

There are two choices for **Physics** in the **Add Component** menu. **Physics 2D** responds a bit differently than **Physics**. Therefore, make sure that you choose **Physics** for this activity!

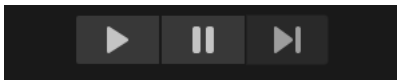
18 A submenu of **Physics** related components should open. These components are useful for many different sorts of behaviors. The one we want is called **Rigidbody**. The **Rigidbody** component allows the **GameObject** to behave like any average object in the real world. Select it to add this component to the bomb object.



19 The **Rigidbody** component has many parameters that can be adjusted to get the **GameObject** to respond to forces inside the game. Right now, we just want to have it respond to gravity, so make sure that **Use Gravity** is checked.

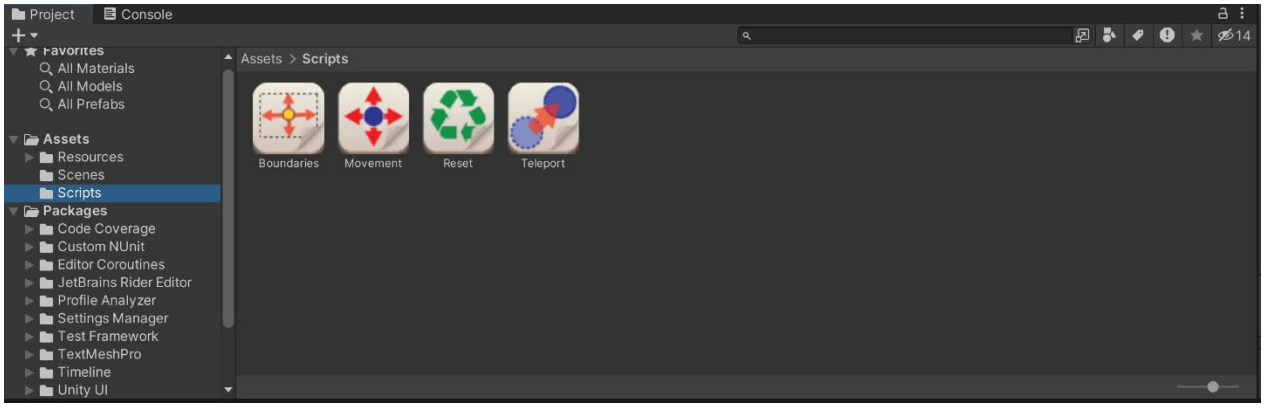


20 Test the game by clicking the Play arrow above the **Scene** panel. The Bomb should fall until it touches the cube and then it stops. When a 3D object like a cube or a sphere is added, a special component called a **Collider** is added with it. Without the Collider, the object wouldn't even know the other object is there and move right through it!

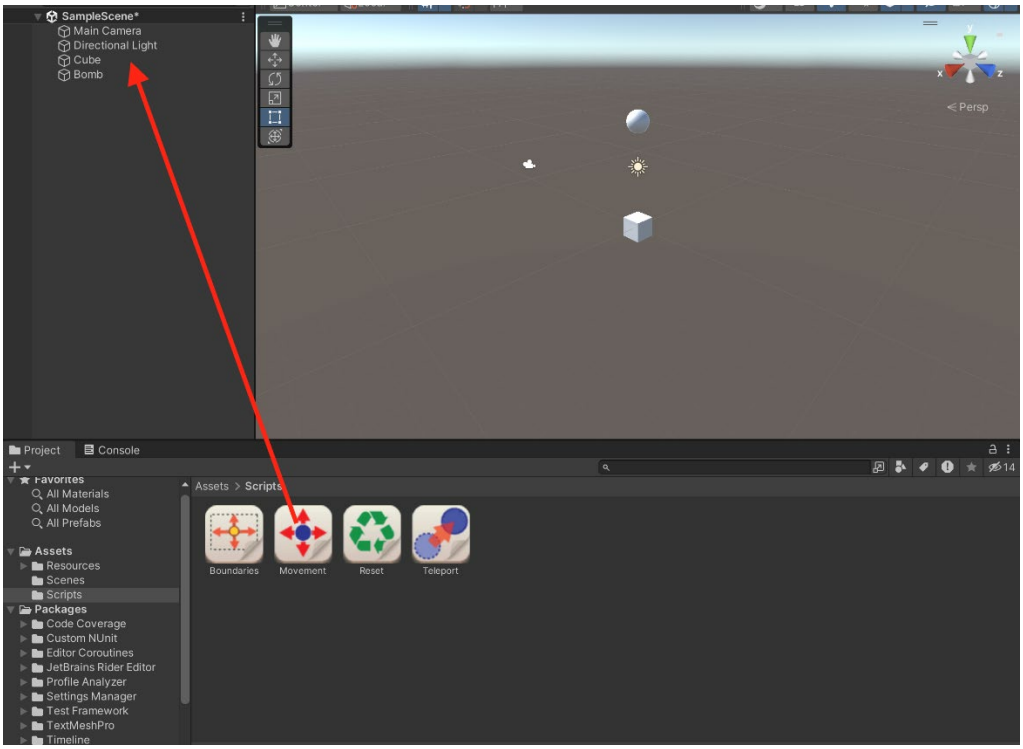


Stop the game by clicking the **Play** arrow a second time.

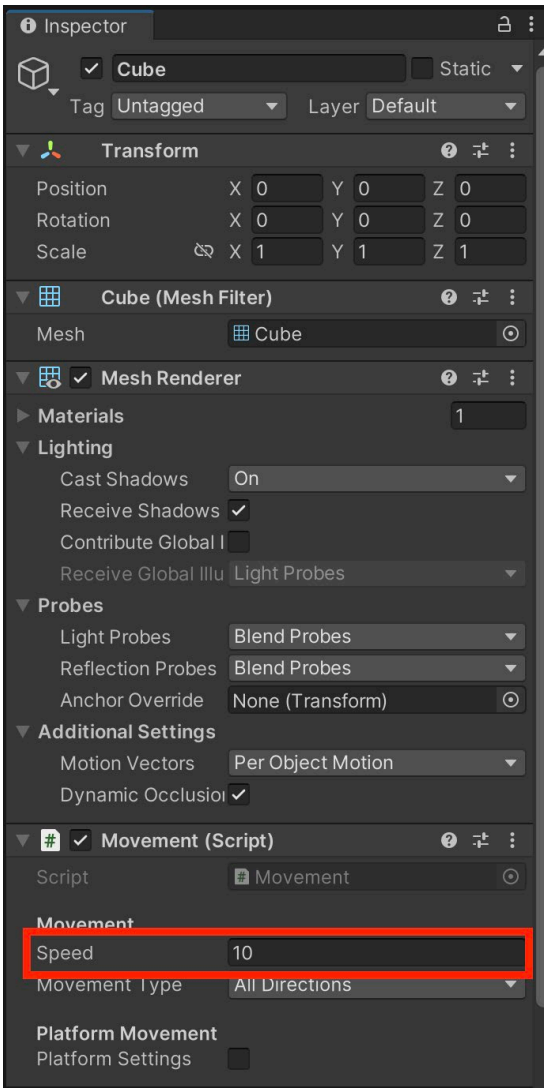
21 The objective of this game is to avoid falling bombs by moving the cube. To move the cube, we'll need a special set of instructions called a ***script***. You loaded some scripts when you imported the unity package. To see them, click on the **Assets** folder in the **Project** panel then click **Scripts** to open the folder.



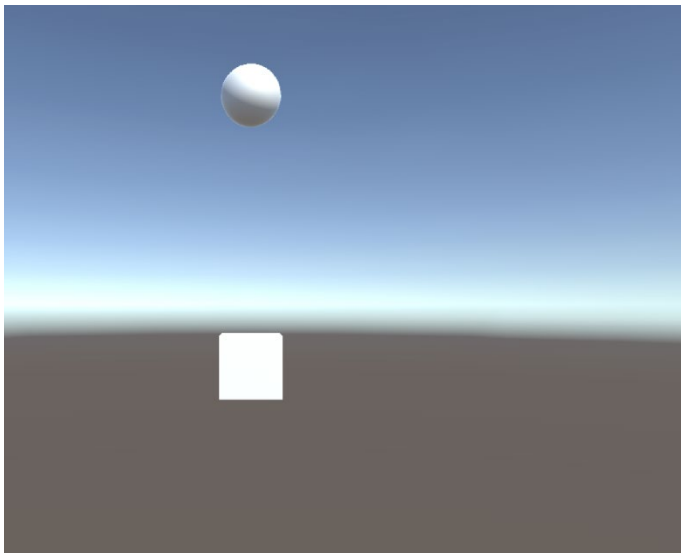
22 Attach the **Movement** script to the cube by simply dragging it into the Cube object in the **Hierarchy** panel.



23 Make sure the cube is selected in the hierarchy panel. You will find the **Movement** script is now part of the components attached to the cube in the **Inspector** panel on the right. By default, the script has a movement speed of **0**, meaning the cube won't move at all. Let's try a movement speed of **10**. Leave the other parameters unchanged.



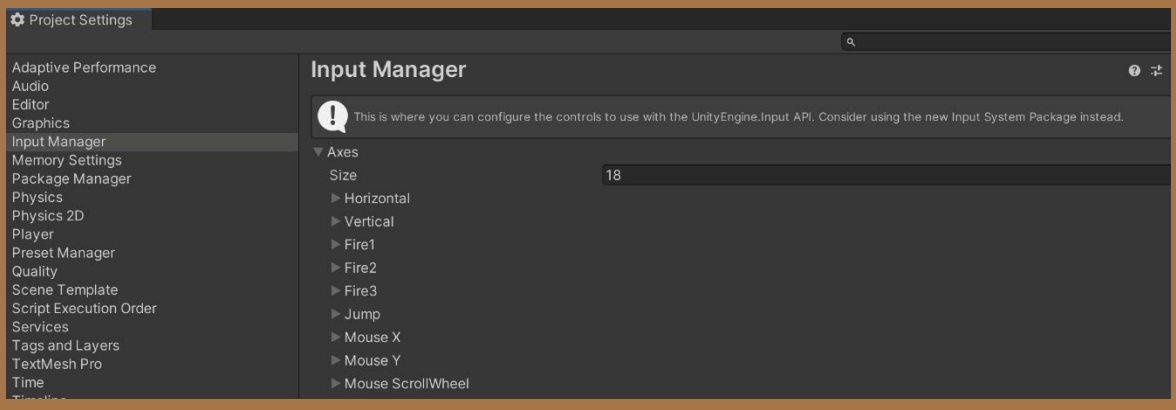
24 Play the game. Use the **arrow** keys or the **WASD** keys to move the cube around. Stop the game by clicking the **Play** arrow a second time.



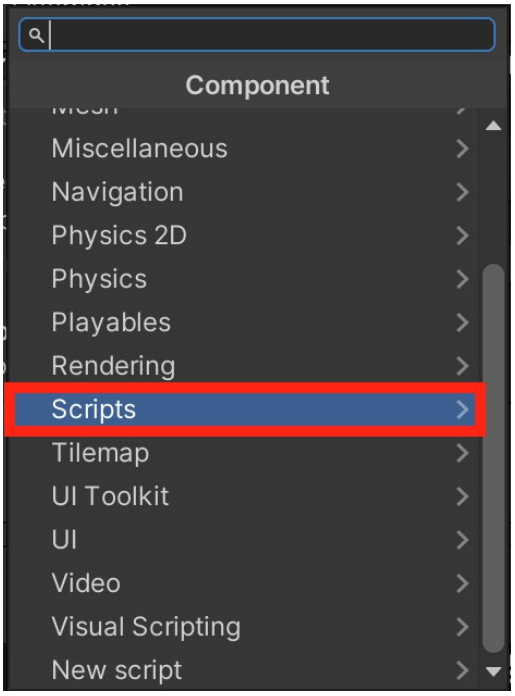
PROJECT SETTINGS

Every project in Unity has a variety of settings that can be configured for whatever situation that you need them for. One of these settings is Input, controlling how the user interacts with the project. Every new project starts out with default settings for input with room for more if needed.

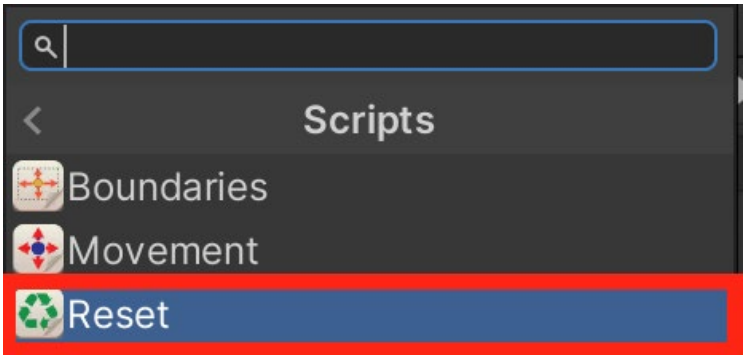
Our provided movement script takes advantage of those settings so that the game can accept input from the arrow keys as well as WASD keys and even a gamepad if one is available!



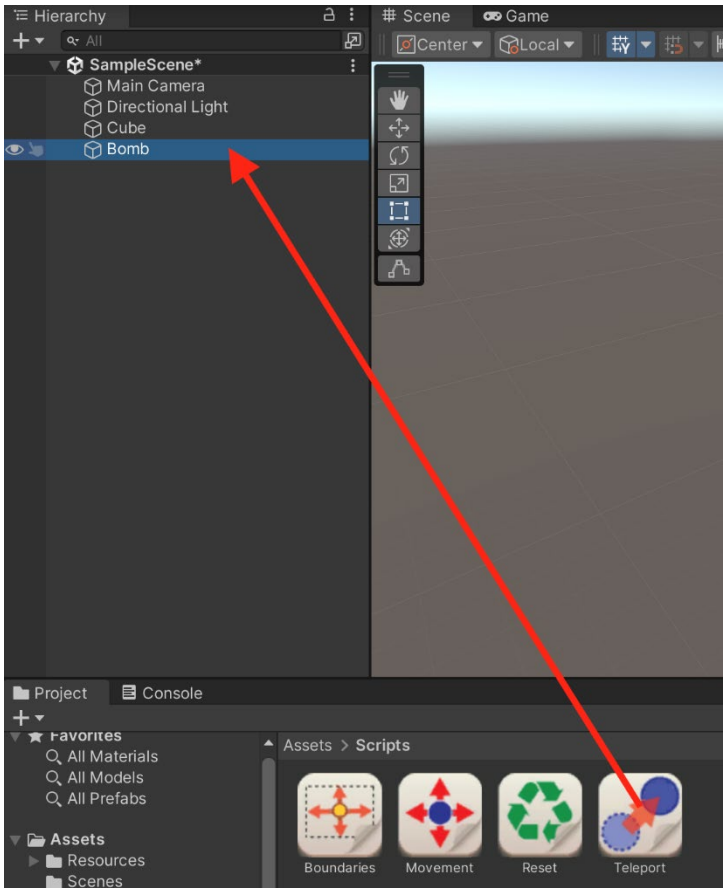
25 To make things more interesting, let's have something happen if the bomb touches the cube. We have a script called **Reset** that will start the game over again if anything collides with the **GameObject** the script is attached to. We want to attach this script to the bomb. Instead of dragging the script over like we did with the movement script, here's another way. Select the bomb in the **Hierarchy** and in the **Inspector** panel, click **Add Component** and select **Scripts**.



26 The menu will change to show the available scripts. Click **Reset** to add the reset script to the bomb.

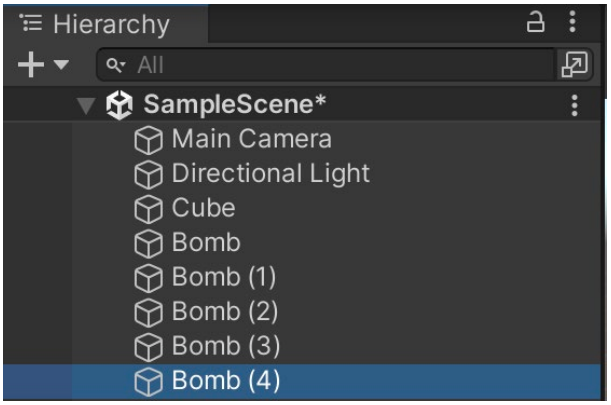


27 Of course, if the bomb doesn't touch the cube, it will just keep falling. We have a script to fix that, too. The **Teleport** script checks to see if the attached **GameObject** has gone below the bottom edge of the screen and if so, places that same object at the top of the screen to let it fall again from a new position. Add the **Teleport** script to the bomb either by dragging it over or by using **Add Component**.



28 To make things more interesting, let's add a few more bombs. Select the bomb in the **Hierarchy** panel and press **Ctrl+D (Windows)** to make a copy. Start with a few, test it, and then add some more until it feels right to you.

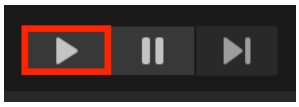
Once you have added some more bombs select your Cube in the **Hierarchy.** In the inspector panel select the **Tag** dropdown menu. In this menu select **Player**.



Pro Tip:

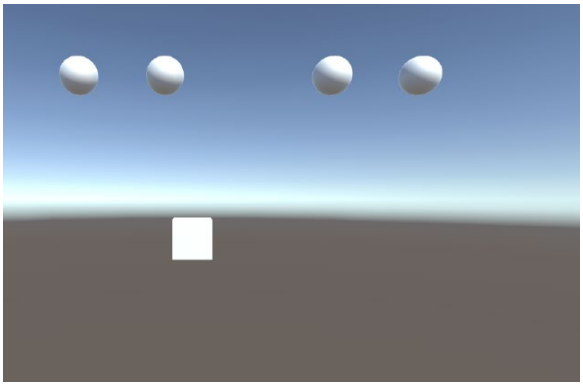
Remember, if you make changes to one of the bombs, you'll need to make changes to the rest of them or delete the other bombs and copy the new one. This isn't very efficient, is it? You'll learn a better way later.

29 Start the game. Use the **arrow** keys or the **WASD** keys to avoid the bombs. If the bomb hits the cube, the game resets to the original start settings. The **Teleport** script is now placing the bomb in random positions on the screen.

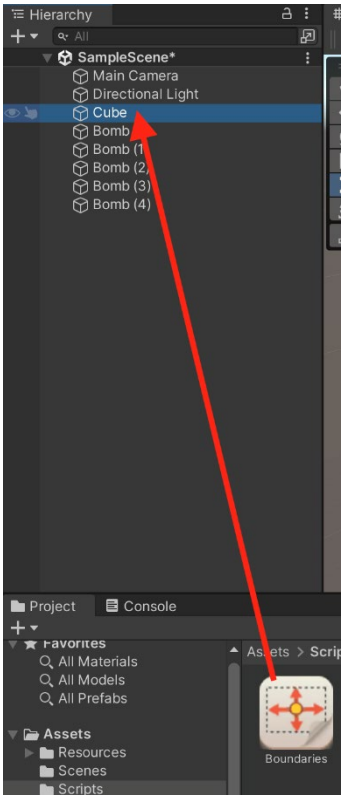


Stop the game by clicking on the **Play** arrow a second time.

30 Now when you start the game, many bombs come from different positions to attack your cube!



31 But there's one last thing to add to the game. At the moment, the cube can avoid all the bombs by leaving the screen! The **Boundaries** script keeps the cube inside the screen. Add the script to the cube to keep it in place using either option learned in this activity.

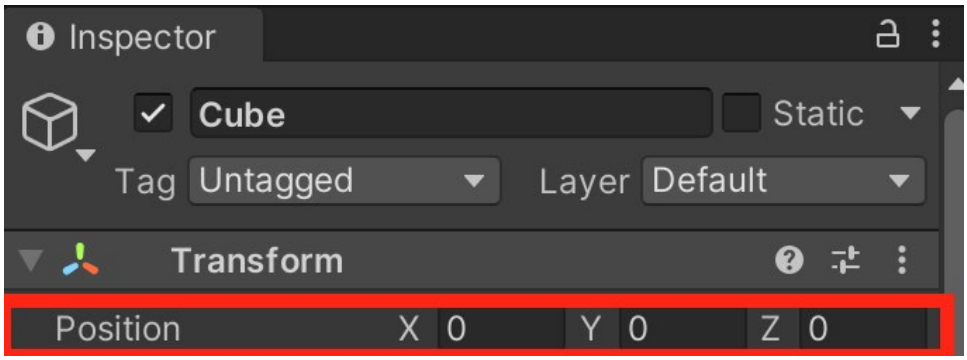


You just made your first game in Unity! Way to go! Later, you'll add some graphics and effects to make it look better, but the best games are fun even before all the graphics are added.

Adjusting the Cube

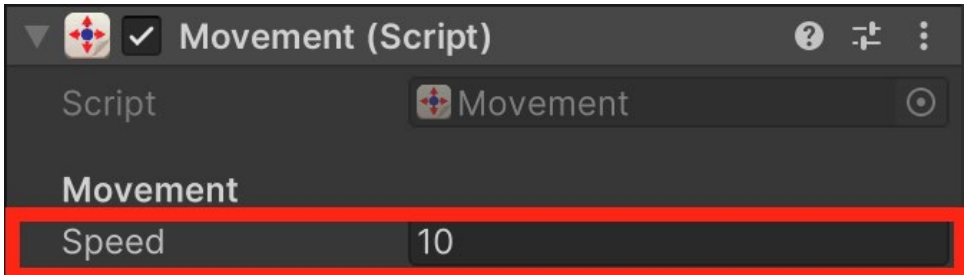
Starting Position

Do you have enough time to dodge the bombs when the game starts? You can move the cube lower in the screen by changing the Y value for the Transform position.



Movement Speed

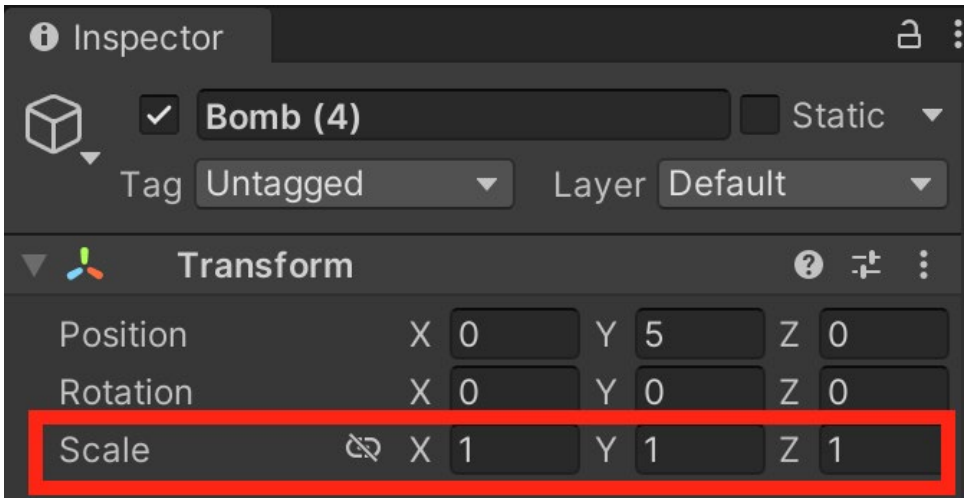
Is the cube not moving fast enough? Increase the number for the movement speed to make dodging easier. Just don't make it so fast that you can't control the cube!



Adjusting the Bomb

Size

Would a smaller target be easier to dodge? Change the scale in the X, Y and Z parameters to a fraction of 1 to make the bomb smaller.



Drag

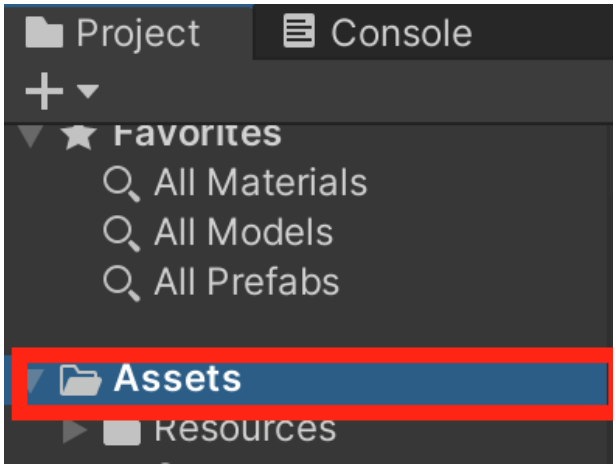
Change the Rigidbody settings of the sphere so that there's a bit of drag to slow it down when falling. Start with small numbers and work your way up.

Don't Forget! When you change the parameters of one bomb it won't change any of the other! Either apply the same changes to all the bombs or delete the unchanged bombs and make copies of the changed one.

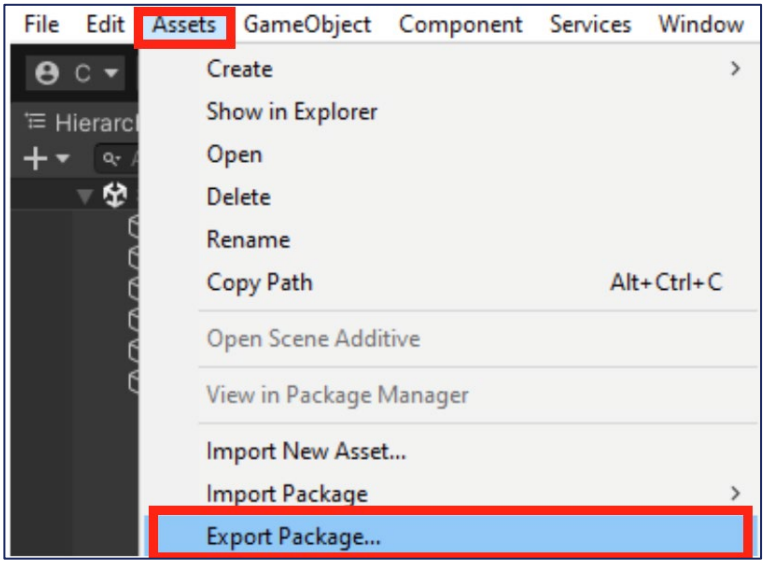
Saving and Submitting Your Games

1 When you save a game in Unity, that project is saved on a folder in your computer for you to open with Unity and resume work on it. However, we need to get it to your Code Sensei so they can grade your game!

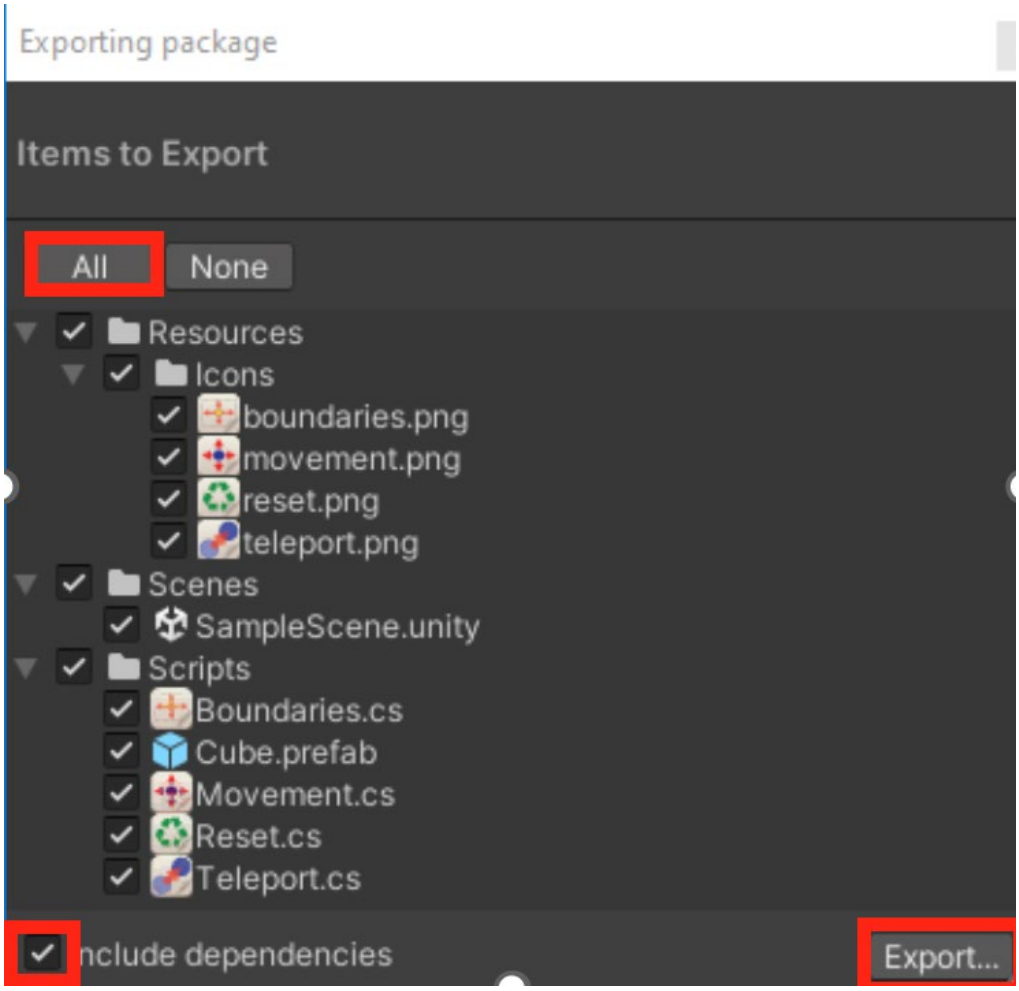
2 In the **Project** folder, click on the **Assets** folder. We do this so that Unity knows we are about to export all the assets and different components that make up your game.



3 We are going to export your game and it will create something called a **Unity Package**. First, click on the **Asset** button in the top left corner then navigate to **Export Package**.

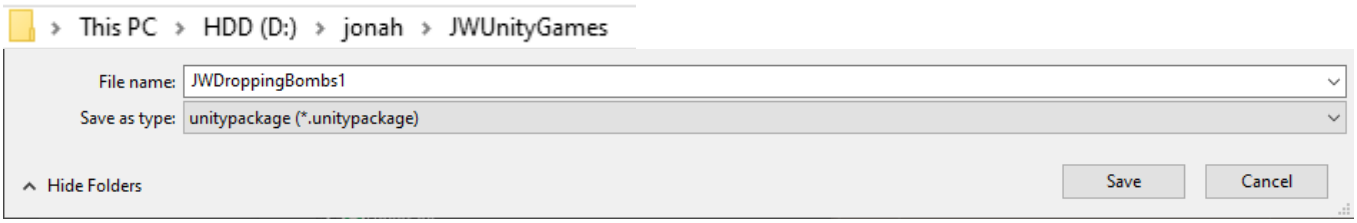


4 Once **Export Package** is selected, the **Items to Export** Menu will appear. Ensure all your game components are checked and the **Include dependencies** checkbox is clicked before clicking on **Export** at the bottom-right.



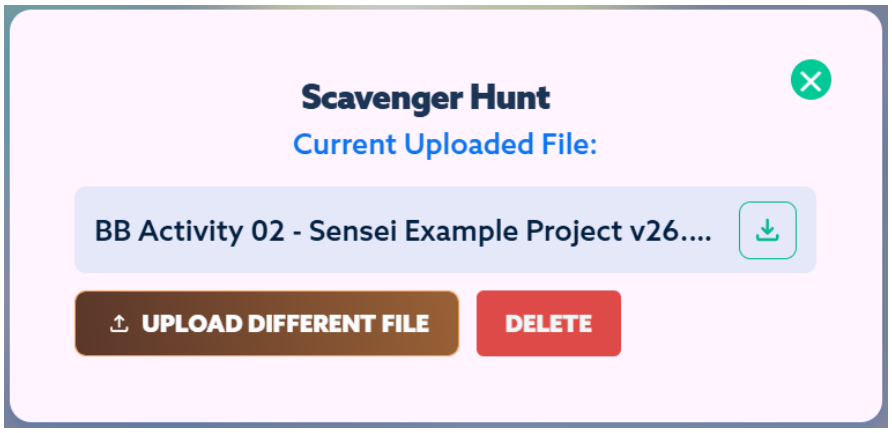
5 Once **Export** is selected, a window will appear asking you where you want to save your game and what you want to name it. This is called a **File Explorer window**. Use a file name that clearly describes your game, such as the Project's name, and put it into a folder that you can find easily.

In this example, we've put it into a folder named "**JWUnityGames**" and named the file "**JWDroppingBombs1**". Your names should include your initials to be a bit more specific!



It may take a minute for the export to finish so just wait until it is complete. Once it is complete you can move onto the next step!

6 In the IMPACT portal, click the **Upload File** button. In the File Exporer window, select the .unitypackage file you just made!



Once the file has been uploaded, close the window. Click the Submit for Review button to

Once you do this, your game is now submitted and ready for grading! You will submit each of your activities like this, so just navigate back to this page if you ever forget how.