Character Controller

[Creating a 3D Game](#_m0kx58bktmys)

[The Standard First Person Character Controller](#_kxy2j934nyjn)

[Importing the Character Controller](#_2kmbrj19y3kr)

[Using the Character Controller](#_x153b1gq36dw)

[Interacting with the World](#_wo5xdyfly7sf)

[Trigger Collisions](#_rn61cnzgshia)

[Collisions](#_jcz2y7ydtm68)

# Creating a 3D Game

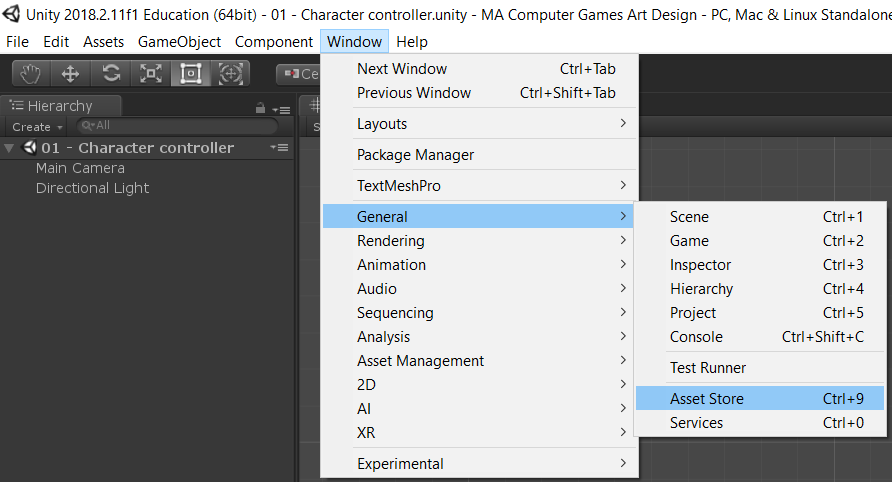
# The Standard First Person Character Controller

Controlling a 3D character in Unity can be a significant challenge, due to the number of issues that needs to be addressed. From the camera movement to the player rigging, each aspect can take hours if not days. Since first and third person shooters are very common, Unity comes with a few simple character controllers ready to be used. Unless your game has any specific requirement, it is strongly advised to start with them.

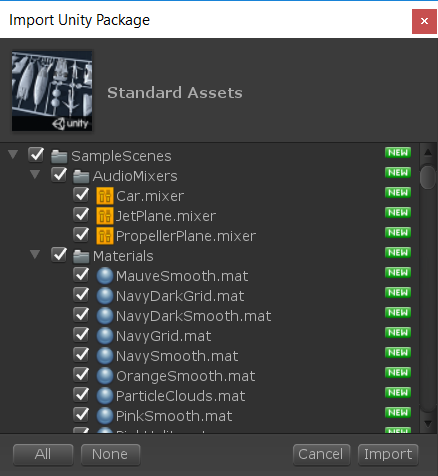
## Importing the Character Controller

The first person character controller was originally available directly from Unity. It is not accessible only through the *Asset Store* in the [Standard Assets](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-32351?aid=1100l45Ay&utm_source=aff) package.

To access it, you first need to open the **Asset Store** from **Windows > General > Asset Store**. Then, search for “*Standard Assets*” and select the one created by “*UNITY TECHNOLOGIES*”.



Assets need to be downloaded, and then they are ready to be imported. This might take a while.



This will create two folders: *SampleScene* and *Standard Assets*, which will significantly increase the size of your project. When building the game, Unity will only include the files that are strictly necessary. This means that the overall size of your game will not be inflated too much.

|  |
| --- |
| **⚠ Paid vs Free Assets**  Not all assets that you can find on the *Asset Store* are available for free!  Within reasons, you are welcome to use any assets that might fit your game. However, you will *never* need to buy anything for this module. Quite the opposite, creative solutions are generally very well appreciated in this course.  When developing games at a professional level, however, relying on paid assets is common as it might help to reduce the overall development time. |

|  |
| --- |
| **💰 Standard Assets**  The [Standard Assets](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-32351?aid=1100l45Ay&utm_source=aff) are a series of assets designed to allow developers creating fast prototypes. These assets were originally available in Unity, but now they are downloadable for free from the *Asset Store*.  The [Standard Assets](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-32351?aid=1100l45Ay&utm_source=aff) package contains a few examples to get you started with:   * First Person Character Controller * Third Person Character Controller * Car Controller * Aircraft Controller * Particles * Rollerball Controller * 2D Platformer * Camera Rigs * Cross Platform Input * Pathfinding |

## Using the Character Controller

To use the First Person Controller, we need to include a prefab called *FPSController*. You can search for that specific asset using the search box on the top right of the **Project** window.



Once located, drag the *FPSController* prefab into your scene. Since it already contains a camera, we will need to delete the existing *MainCamera* game object present in the scene.

Once that is done, the first person controller should be working and react to mouse, keyboards and controllers as expected.

|  |
| --- |
| **💡 RigidBodyFPSController**  The FPSController uses a script called FirstPersonController which implements its own physics. This means that while it has a Rigidbody component, it does not react to physically collisions as you would expect.  If your character needs to be affected by forces and to react to collisions in a realistic way, then you can another prefab called RigidBodyFPSController. This one has all the expected behaviours, but offers fewer features compared to FistPersonController. |

|  |
| --- |
| **📖 Character Controller**  The FirstPersonController component relies on a special type of collider which is called CharacterController. Despite the name, the CharacterController component is indeed a collider and it has the shape of a capsule. The shape has been chosen as it can smoothly move across a variety of surfaces. |

|  |  |  |
| --- | --- | --- |
| **Prefab** | **FPSController** | **RigidBodyFPSController** |
| Script | FirstPersonController | RigidBodyFirsPersonController |
| Collider | CharacterController | CapsuleCollider |

# Interacting with the World

The standard first person character controller can be used to interact with the world in many different ways. If you need to detect collisions with the players, it is common practice to tag the FPSController with the “*Player*” tag.

## Trigger Collisions

If you want to detect the presence of the player in a specific region, you can use a trigger collider with a script that implements the following method:

|  |
| --- |
| void OnTriggerEnter (Collider collider)  {  if (collider.CompareTag(“Player”))  {  // The player has entered the trigger area!  }  } |

## Collisions

Even if a Rigidbody component is present on the FPSController prefab, neither the FirstPersonController script nor the CharacterController component rely on rigidbody physics.

By design, the CharacterController can be detected using trigger collision methods (OnTriggerEnter, OnTriggerStay, OnTriggerExit), but it will not react to physics collision (OnCollisionEnter, OnCollisionStay, OnCollisionExit).

Physical collisions can be detected by using the method OnControllerCharacterHit in a script placed on the FPSController gameobject:

|  |
| --- |
| void **OnControllerColliderHit** (**ControllerColliderHit** hit)  {  // The player has hit a collider  } |

The table below shows, in the event of a collision between the FPSController and a gameobject, which methods can be used to detect collisions.

|  |  |  |
| --- | --- | --- |
|  | FPSController | Other GameObject |
| **Trigger Events**  OnTriggerEnter  OnTriggerStay  OnTriggerExit | ✔️ | ✔️ |
| **Collision Events**  OnCollisionEnter  OnCollisionStay  OnCollisionExit | ❌ | ❌ |
| **Controller**  OnControllerCharacterHit | ✔️ | ❌ |