In this guide we will be

We will use the language GDScript, though all the functionality can also be done in C#.

# Step 1 - Creating a Project

A screenshot of a computer

Description automatically generated

Ignore a pop-up window asking to open the Asset Library as you have no projects.

Click ***+*** *New Project* in the top right in Godot 4 or ***+*** *New* in the top left in Godot 4.2.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

Name the project in *Project Name* and navigate to the folder you want to save the project in

Click *Create Folder* to add an empty folder in the *Project Path* folder with the name of your project. The project will be placed in this created folder.

# Step 2 - Importing Assets

Start by dragging the Environment folder from the Windows File Explorer into the *FileSystem* window in Godot. In the *Base* folder there are many models which can be used to build a level.

# Step 3 - Creating the Level Scene

A screenshot of a computer

Description automatically generated

In the *Scene* panel, select *3D Scene* (or *2D Scene* if you prefer). This will be the root node of the level scene.

A screenshot of a computer

Description automatically generated

Double-click on the created node in the *Scene* panel to rename it, e.g. to *Level*.

Save the scene as a *.tscn* file, e.g. *Level.tscn*

A *.tscn* is a scene file represented in text, while a *.scn* is represented in binary. A *.tscn* scene is useful for debugging.

A screenshot of a computer

Description automatically generated

Click the **+** in the *Scene* tab to add a *StaticBody3D*, and a *CollisionShape3D* and *MeshInstance3D* as child nodes.

A screenshot of a computer program

Description automatically generated

While selecting the *CollisionShape3D* in the *Scene* hierarchy, in the *Inspector* add a *BoxShape3D* to the *Shape* property of the *CollisionShape3D*. Set its *Size* to be wider, e.g. 10, 0.5, 10.

A screenshot of a computer

Description automatically generated

Add a *BoxMesh* to the *MeshInstance3D* with the same *Size* as the collision *BoxShape3D*.

The y position of the *StaticBody3D* can be set to negative half of the height of the box such that the surface is at y = 0.

A screenshot of a computer

Description automatically generated

Click the three vertical dots along the panel above the viewport to add a default *DirectionalLight3D*, and a default *WorldEnvironment*.

We now have a basic level to place a player in. Drag and drop the *Player.tscn* from the *FileSystem* into the *Scene* tab or the *Viewport* of the Level scene.

# Step 3 - Creating the Player Scene

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

In the *Scene* panel, select *Other Node* and search for *CharacterBody3D*.

A screenshot of a computer

Description automatically generated

Click the **+** in the *Scene* tab to add *CollisionShape3D* and *MeshInstance3D* nodes. Double-click on a node to rename it, e.g. *CharacterBody3D* to *Player*.

A screenshot of a computer

Description automatically generated

Selecting the *CollisionShape3D*, in the Inspector add a *CapsuleShape* to the *Shape* property.

Similarly add a *CapsuleMesh* to the *Mesh* property of the *MeshInstance3D* with the same dimensions as the collision shape.

A computer screen shot of a white cylinder

Description automatically generated

You should now have a *Player* scene like this.

A screenshot of a computer

Description automatically generated

Next select the Player root node (*CharacterBody3D*) and in the *Inspector* click where it says *<empty>* in the *Script* property.

In the script creation window, you can select the language, either GDScript or C# if using the .NET version of Godot. In this workshop we will be using GDScript. The script should inherit *CharacterBody3D* and use the *Template* for *Basic Movement*.

Name the script, e.g. *Player.gd* and create a *Scripts* folder to save it in.

A screenshot of a program

Description automatically generated

Click Create and you will see the *Basic Movement* template in the *Script* tab.

Add a *Node3D* for the head of the player and a *Camera3D* as a child node.

A screenshot of a computer

Description automatically generated

If you like, the *CollisionShape3D* and *MeshInstance3D* can be moved up half of the height such that the bottom is at the scene origin.

The *Head* can be moved up to align with a suitable head height on the capsule.

At the top of the script, add these lines to the Player.gd script. The annotation *@onready* means the variable assignment will automatically occur during the *\_ready()* function.

A close up of a logo

Description automatically generated

Create an *\_input(event)* function in which we will rotate the head and camera according to mouse motion.

A screen shot of a computer program

Description automatically generated

This rotates the head according to the horizontal mouse motion and rotates the camera according to the vertical mouse motion.

A screen shot of a computer program

Description automatically generated

We can clamp the vertical camera rotation between 90° and -90° so the player can’t rotate the camera past vertically up or down.





At the top of the script, add a float variable called *look\_sensitivity* and multiply the head and camera rotation by it.



Add two float variables *moving\_lerp\_speed* and *stopping\_lerp\_speed* to control the lerp when moving and stopping at different speeds.

A screen shot of a computer code

Description automatically generated

Use the built-in lerp function in the movement code and the two lerp speed values.