CODE NINJAS®

Bronze Belt Ninja Guide

Activity 00: Welcome to Godot

Working in Godot

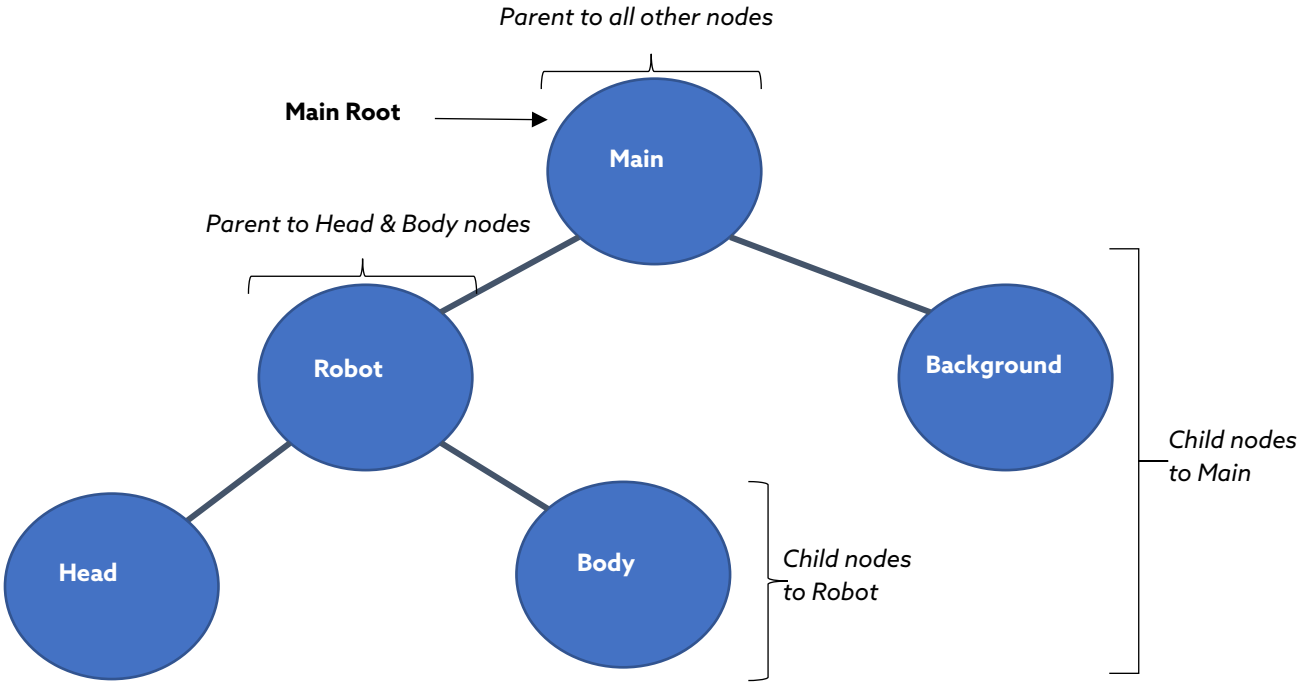
To begin, you will explore the Godot Interface by learning how to find project components and how to modify them. For the first few activities, you will use pre-built scripts to create each game, easing you into Godot's environment. Once you are familiar with it, you will start working with Godot's **GDScript**.

Godot has its own scripting language called **GDScript**. Its syntax is very similar to another popular coding language you may have heard of or tinkered with before – **Python**. Check out some examples of games below that were made using Godot!

Image Sources: *Sonic Colors: Ultimate, Dome Keeper, Cassette Beasts*

Using Nodes

Previously, you worked with Sprites in IMPACT or Game Objects in Unity to create games. In Godot, **nodes** are the building blocks of a game. They are objects which store all different kinds of data, such as sprites, functions, background images, animations, and more! Nodes are arranged in a **tree-like hierarchy**, as modeled below. Trees are a common **data structure** used in computer science. Data structures are ways to store and organize data.

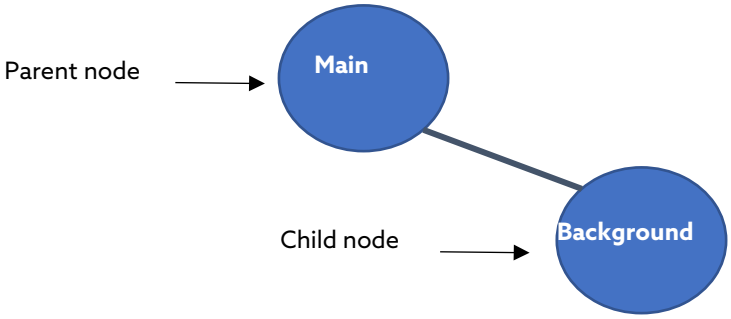


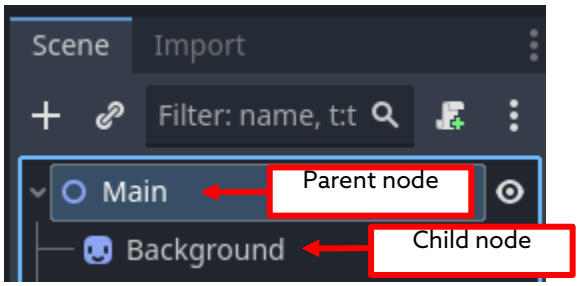
To start a project, a **Main root** will need to be created. Think of this like the root of a tree. The **Main root** is the **parent** to all other nodes in your project. If a node is a parent, it has at least one **child** node; think of a child node as a branch in a tree. The rest of the nodes in your project are child nodes that connect to the **Main root**. Each node can have multiple child nodes, and these child nodes can also have their own child nodes. That can create a pretty big tree!

A tree of nodes can be saved into a file called a **scene**. Godot requires a project to have a **main scene** to run the game. This **main scene** simply tells Godot where to start the game, so it is usually saved from the **Main root** node.

Parent and Child Node Relationships

Child nodes are dependent on their parent nodes and can inherit some of their properties. When a parent node is changed, it affects all its children. For example, if a parent node is deleted, all children will also be deleted. If a parent node's position is changed, each child node's position changes. This relationship makes it easier to keep the game organized and makes objects behave together as a group.





Notice the tree above. On the right side is an example of what nodes and parent-child relationships look like in the Godot editor.

Sensei Stops

Pause for a **Sensei Stop!**

Throughout these projects, there will be **Sensei Stops.** A **Sensei Stop** is a checkpoint with a **Code Sensei** to make sure everything in your game is correct up until that point before moving on. Think of it like the yellow "Check Code" buttons in IMPACT. This will help with debugging projects and give you a chance to show off your amazing hard work!

At each **Sensei Stop**, make sure to:

- Pause and call over a Code Sensei.
- Review previous steps and concepts, while waiting for a Code Sensei.
- Show off the game and explain what you understand to the Code Sensei.
- Get the OK from a Code Sensei before continuing onto the next steps.

At this **Sensei Stop**, make sure to check in with a Code Sensei to tell them what you know about parent and child node relationships!

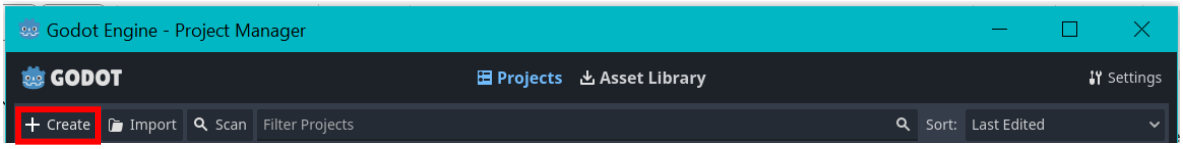
Activity 00 – Welcome To Godot

Time to level up your skills with Godot! Godot is a powerful game engine that enables you to create advanced and customized games. In this activity, you will explore the Godot interface through an interactive tour. After the tour, you will start building your very first project in Godot by tinkering with the editor's tools. By the end of this activity, you will know how to navigate the Godot editor, organize your future projects, import files, reparent nodes, and map custom inputs.

1 Open **Godot** and view the **Project Manager**.

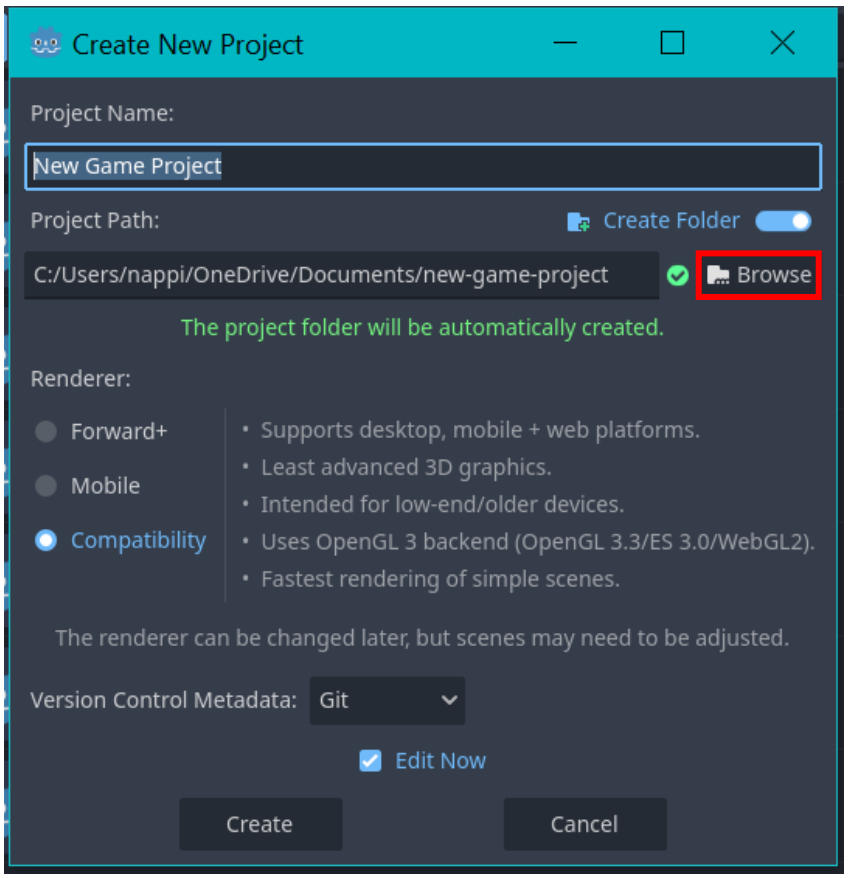
This is where all Godot games can be viewed.

Select the + **Create** button in the top left to create a new project.



2 Store all Godot projects in the same place.

Select **Browse.** This will create a pop-up window to select where the project will be saved.



Pause for **Sensei Stop #1!**

Check in with a Code Sensei to select the correct **Project Path** to save all Godot games.

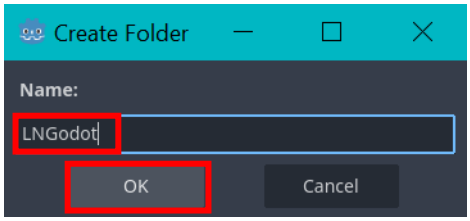
A **Project Path** is where projects will be saved on the computer. The / indicates the project is saved inside of a folder.

Pay attention to the selected **Project Path**; all future projects will be stored in the same place!

3 Create a personal folder to store each Godot project!

At the top right, select the **New Folder** icon and a **Create Folder** window will appear.

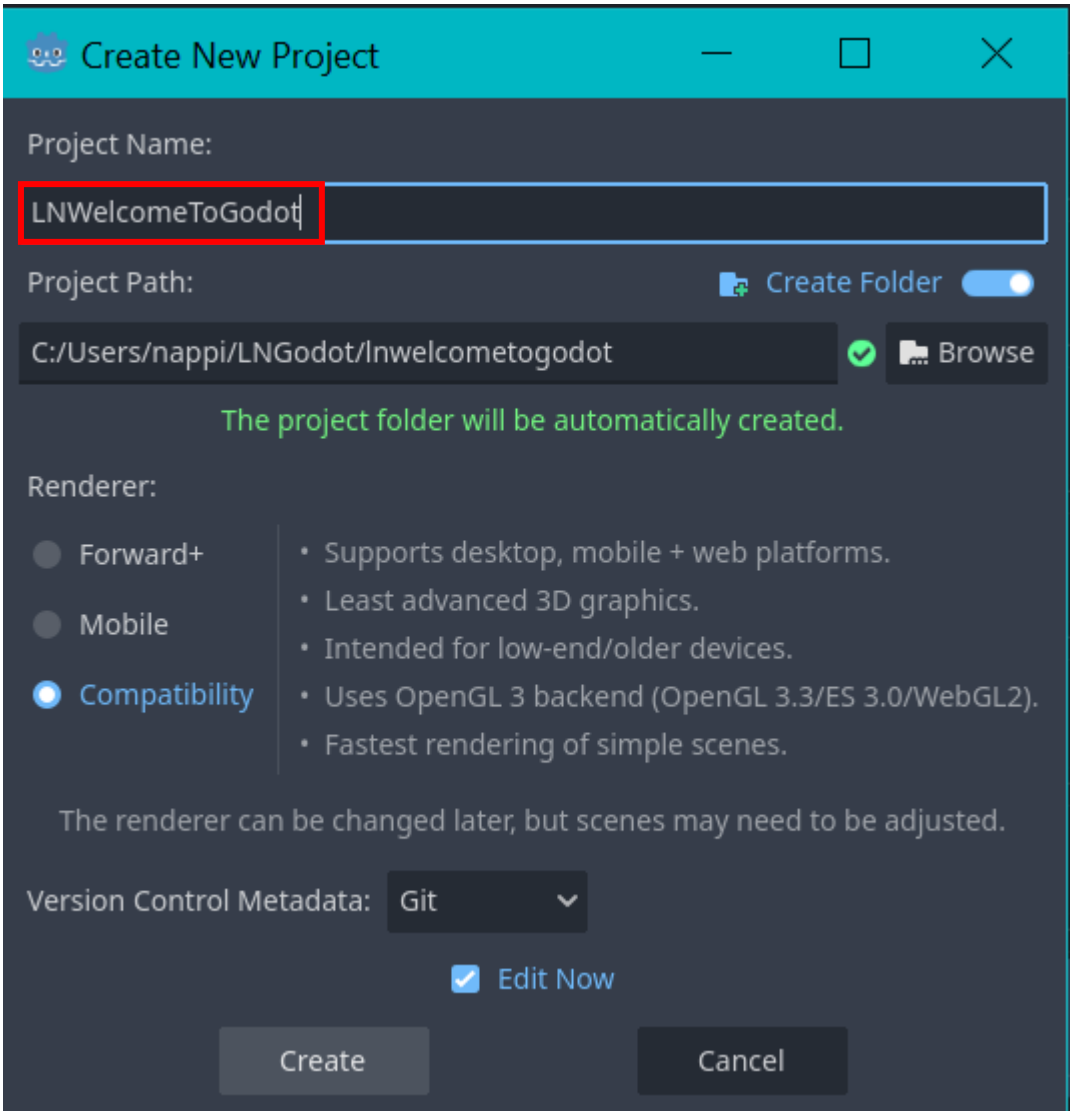
4 Name the folder **MyInitialsGodot.** For example, the folder name could be "LNGodot." Click **OK.**



At the bottom, click **Select Current Folder.**

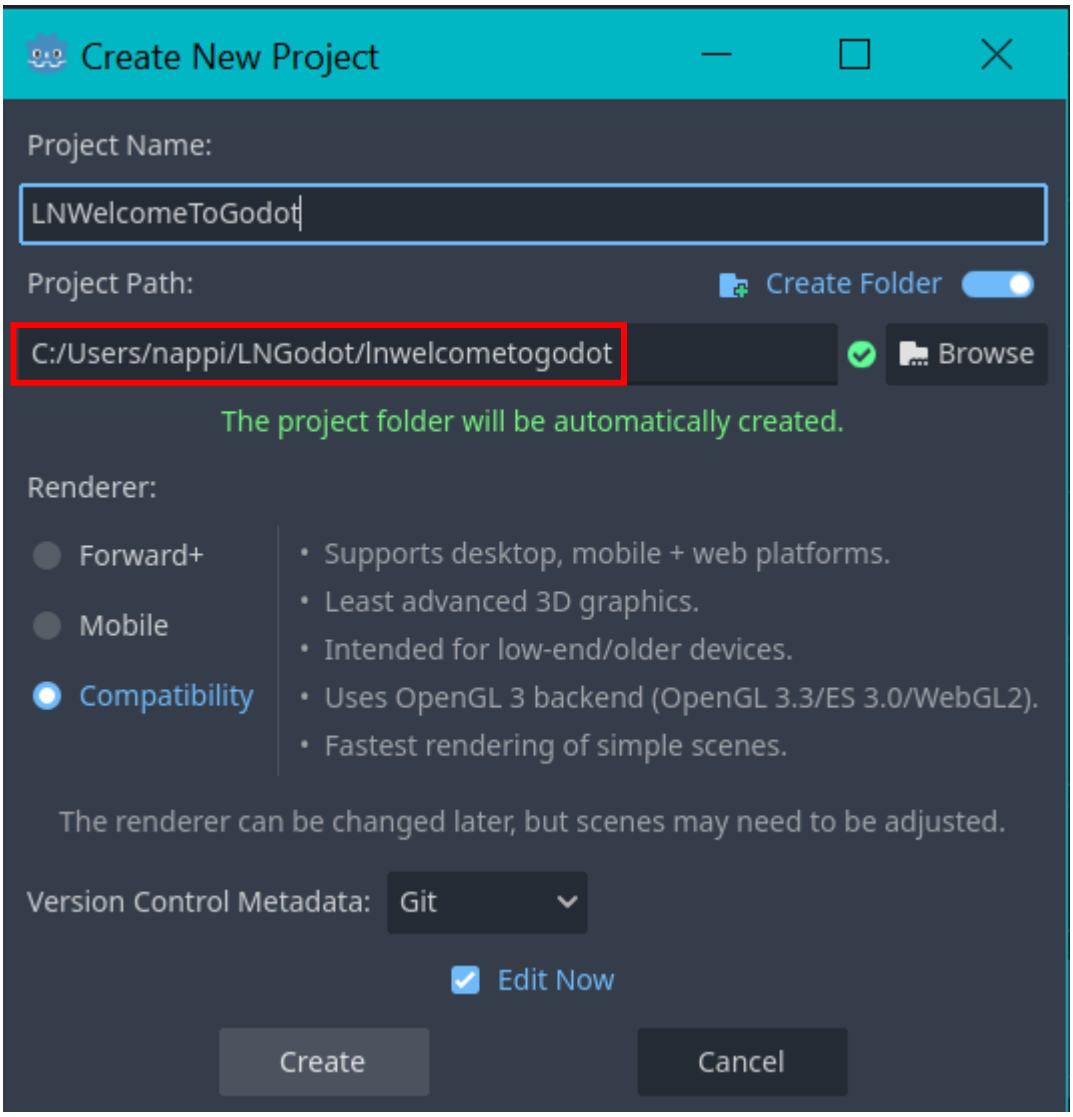
5 All **project names** will include your initials and the name of the activity.

Rename the project: **MyInitialsWelcomeToGodot.** For example, the project name could be "LNWelcomeToGodot."



6 Take note of the **Project Path**; this is where all Godot projects will be saved. Selecting the right **Project Path** is very important. If a game is stored in a different place or another Ninja's folder, it could get lost!

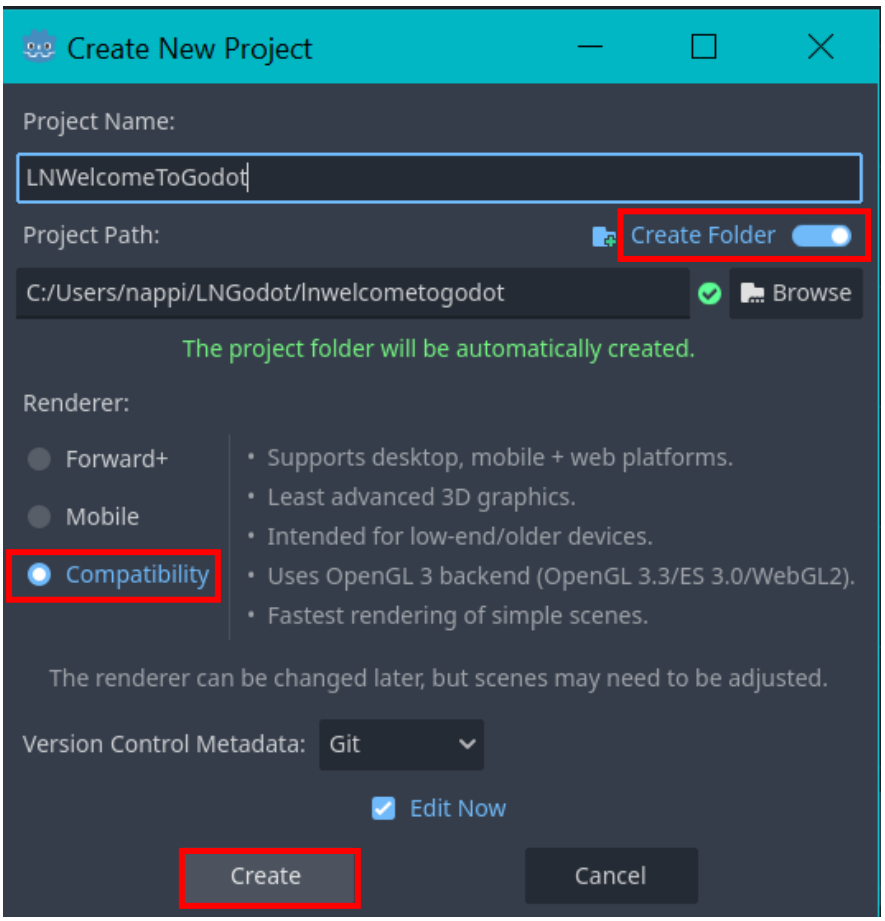
The project path shown might be slightly different from what the **Code Sensei** selected and that's okay.



Before creating any project, always make sure the **Project Path** is correct to avoid losing projects. Don't be afraid to ask a **Code Sensei** for help with this.

7 Make sure **Create Folder** is toggled **on** to keep things organized.

Select **Compatibility** as the renderer, then click **Create**.



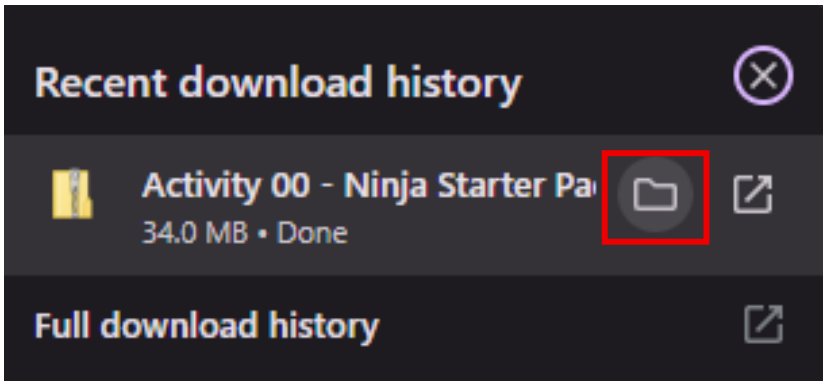
Pause for **Sensei Stop #2!**

Check in with a Code Sensei to locate the shared file location. A shared file location will store all assets that will be downloaded and imported for future projects!

For this project an **interactive tour** will need to be imported to explore the Godot editor.

Download **BB Activity 00 – Ninja Starter Pack.zip**

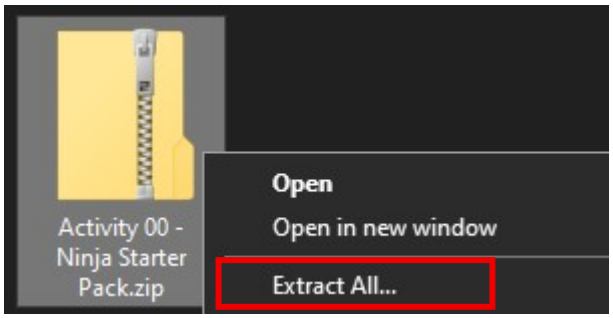
8 After downloading **BB Activity 00 – Ninja Starter Pack.zip**, a **Downloads** window should pop up at the top right of the screen. Click on the **folder icon** to open the location of the download.



If the **Downloads** window disappears, click **CTRL + J** on the keyboard to bring it back. Click the folder icon to open its location.

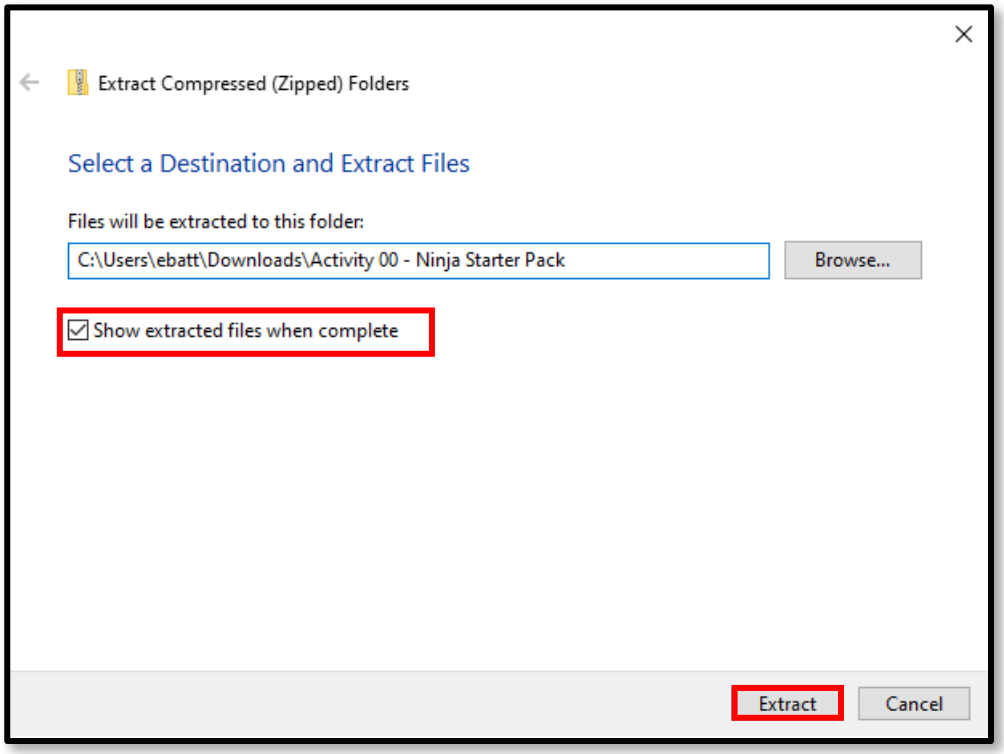
9 Notice the folder is zipped. Files can't be opened when they are zipped like this. To open a zipped folder, it must first be unzipped or extracted.

Right-click on the **zip** file and select **Extract All.**

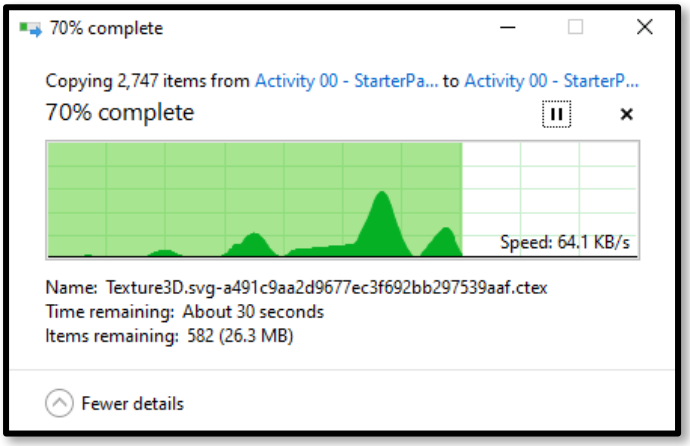


10 This will create a new window to select a place to put the extracted files.

Check **Show extracted files when complete** then click **Extract.** This will create another **File Explorer** window exactly where the extracted files are; do **not** close that window.

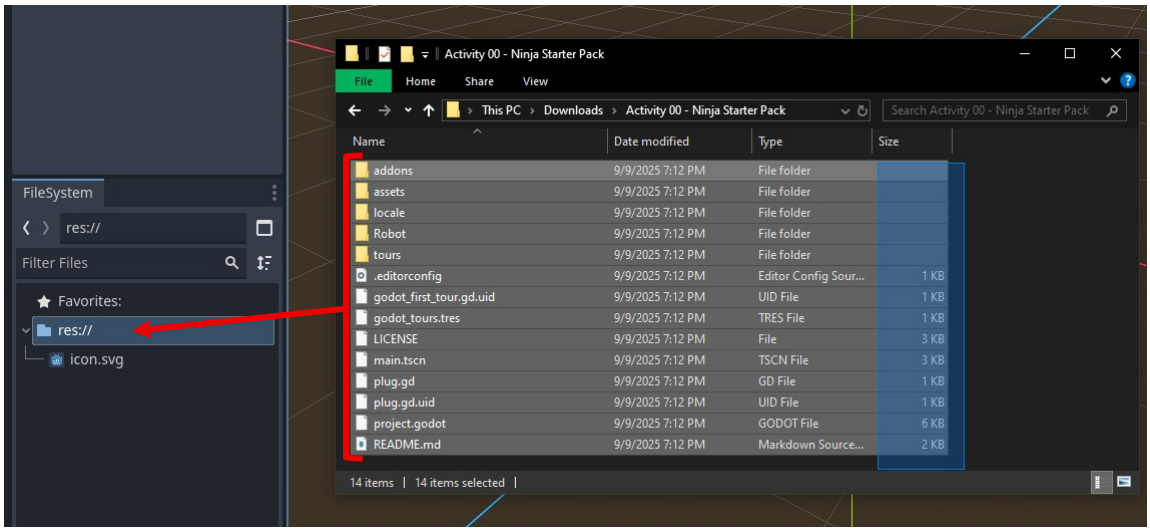


Because all computers have their own username, the path will look slightly different! There are a lot of files to extract for this Tour, so wait a couple minutes for everything to be extracted.



11 Have both Godot and the **File Explorer** with the extracted files open.

Press **Ctrl + A** to select **all** the files, then drag them into **res://** in **FileSystem** as seen in the picture.

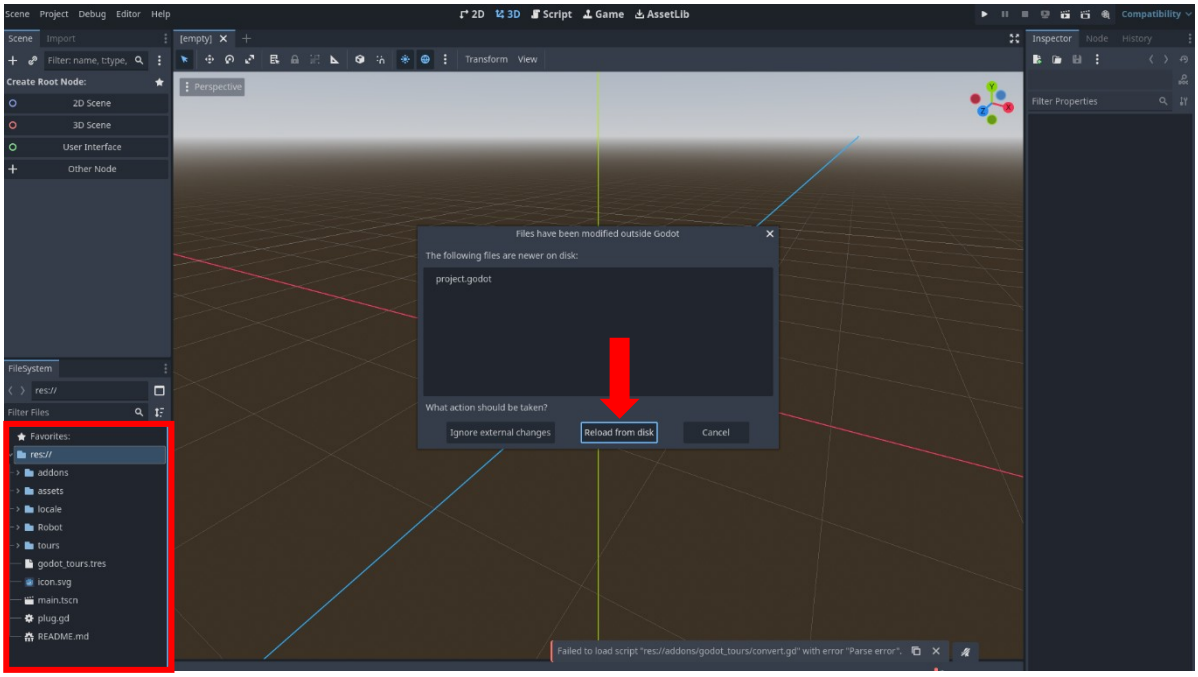


Reminder:

File Explorers can be reopened in the browser by pressing **CTRL + J** on the keyboard and clicking the **folder** icon.

12 Click on the Godot editor and wait several moments for the assets to download.

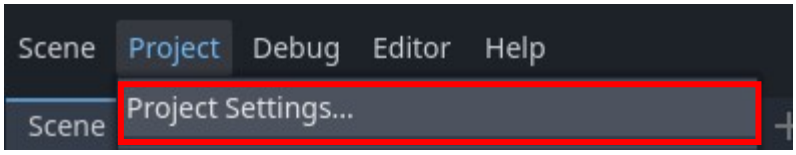
A popup may occur that changes have been made to the **project.godot** file, click **Reload from disk** to accept the new changes.



Ignore the **Failed to load script** error.

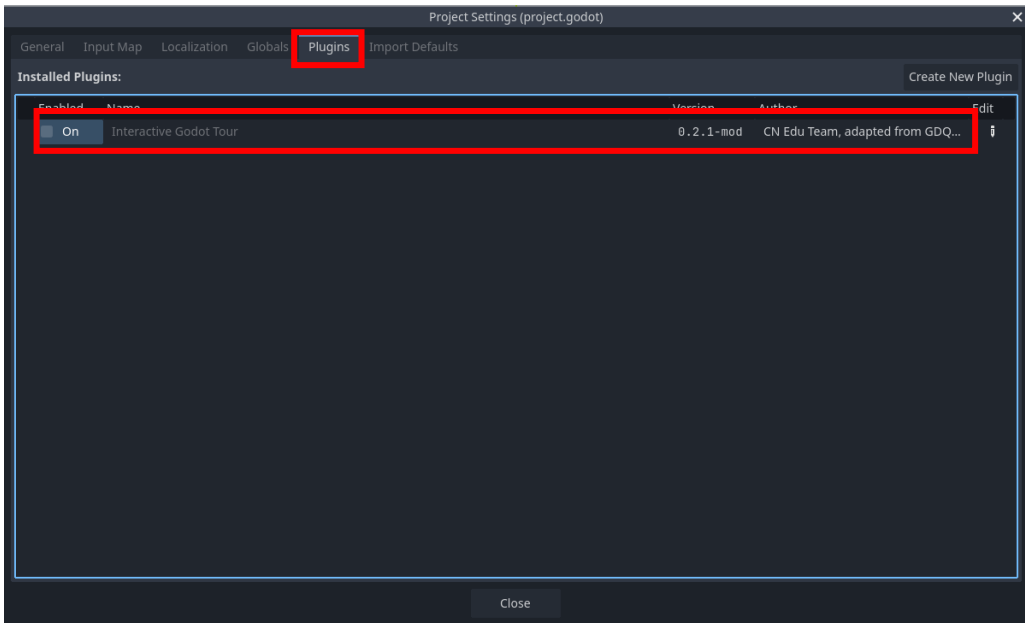
13 The Interactive Tour is a plugin that needs to be enabled.

At the top left of the Editor, select **Project** and click on **Project Settings...**

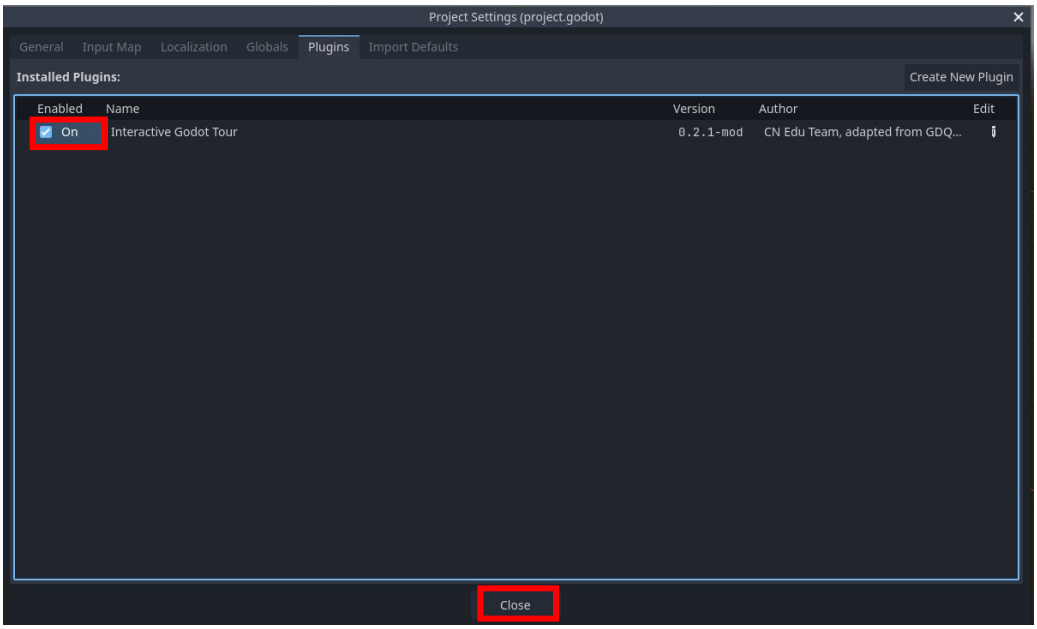


14 At the top of the window, click on **Plugins.**

A **Godot Tour** plugin should be visible. If not, ask a **Code Sensei** for help.



15 Click on **Enable** to enable the Godot Tours plugin.



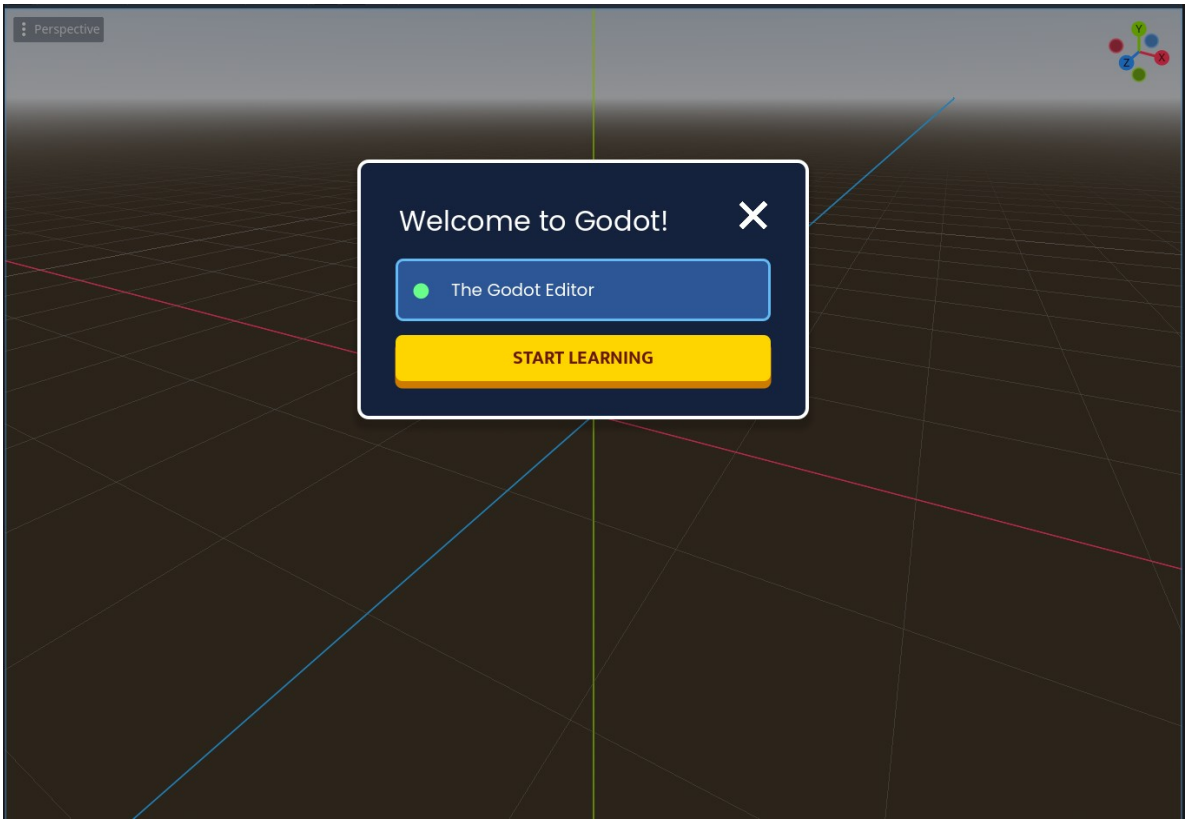
Select **Close** to return to the Editor.

If the Godot app closed, wait shortly as it will reload in a few moments.

16 The interactive tour should now pop up.

Click on **Start Learning** to begin the tour!

Follow the steps to complete the tour. Once finished, return to this **Ninja Guide**.

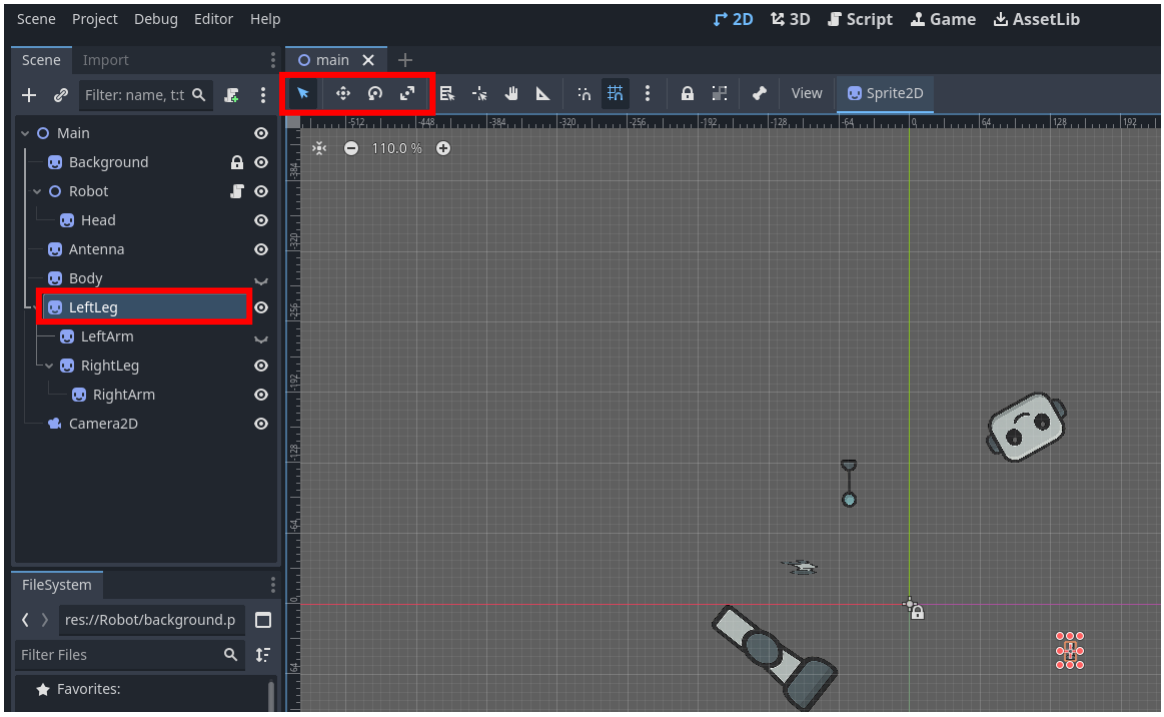


17 Notice the **TOUR** button in the top right of the editor. This button will only be available for this activity and allows the tour to be revisited at any time, if needed.

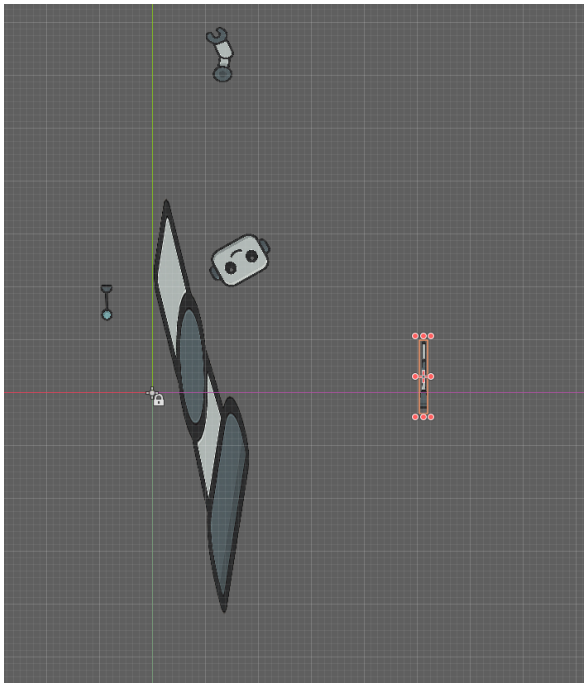
Pause for **Sensei Stop #3!**
Check in with a Code Sensei before moving on.
Point out each part of the Godot editor such as the Scene, Inspector, Viewport and FileSystem.

18 Time to fix the Robot!

Find the **Scene Menu** on the left side of the screen and select the **LeftLeg** node.

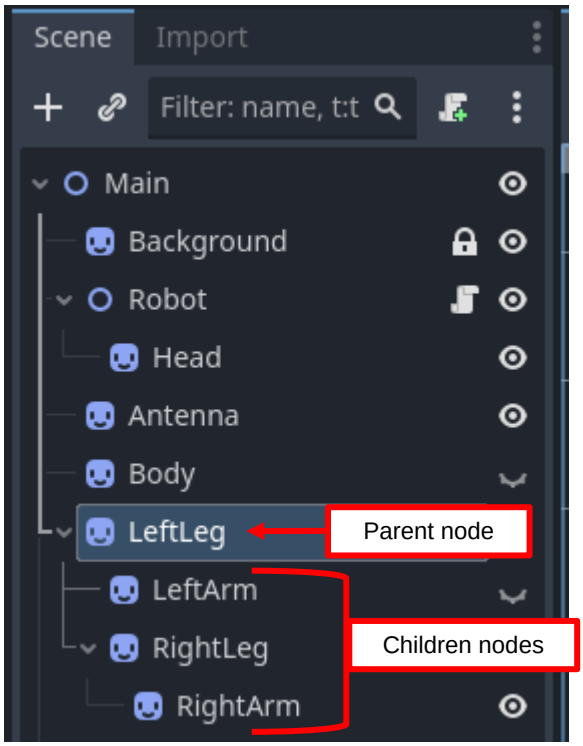


Using the **Tool Bar** from the **Interactive Tour,** try to **move** or **resize** the node. What happens to the Robot parts?



19 It seems when the **LeftLeg** part is moved or resized, other parts of the Robot are changed too!

Revisit the **Scene Menu** and notice the **LeftLeg** node is a **parent** node to the **LeftArm, RightLeg,** and **RightArm** nodes.

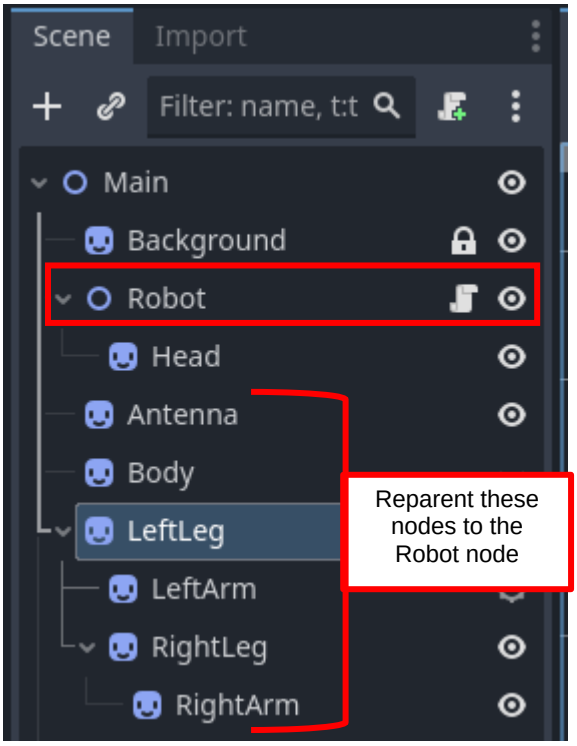


When a parent node is changed, it affects all its children; this is why when the **LeftLeg** is moved, the **child** nodes move too.

Reminder:

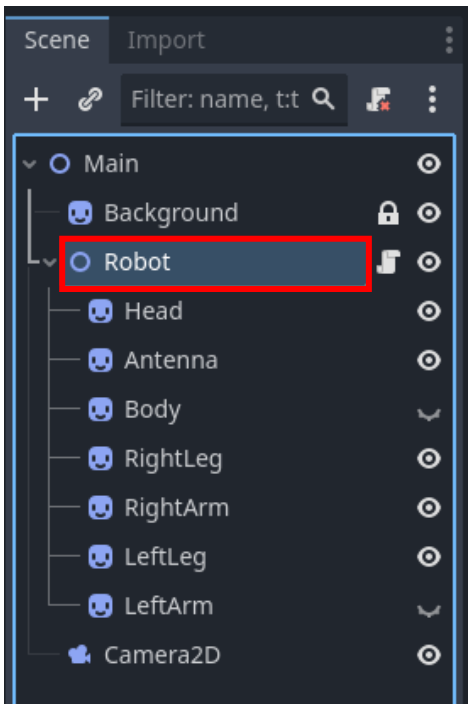
Nodes are organized in a **tree structure**, where some nodes are **parents** and others are **children.** Child nodes **inherit** position, rotation, and scale from their parent.

20 **Reparent** the Robot part nodes in the **Scene Menu**, to make it easier to fix the Robot!



Drag and drop each Robot part node onto the **Robot** node. This will make all the Robot parts **children** to the **Robot** node.

The hierarchy should be the same as shown.



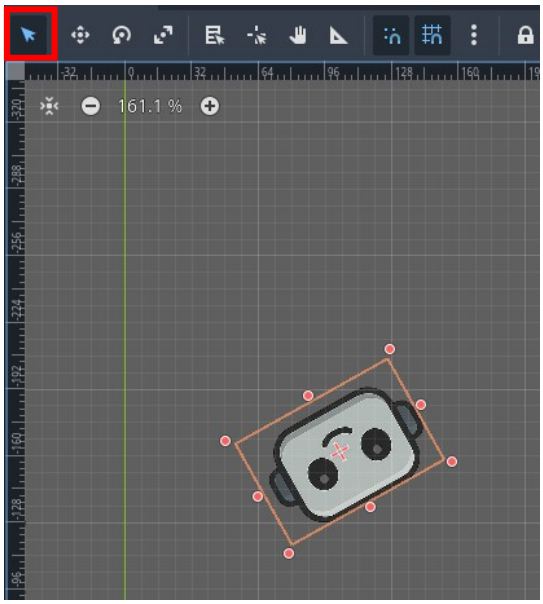
Pause for **Sensei Stop #4!**

Wait for a Code Sensei to check the nodes have been **reparented** correctly.

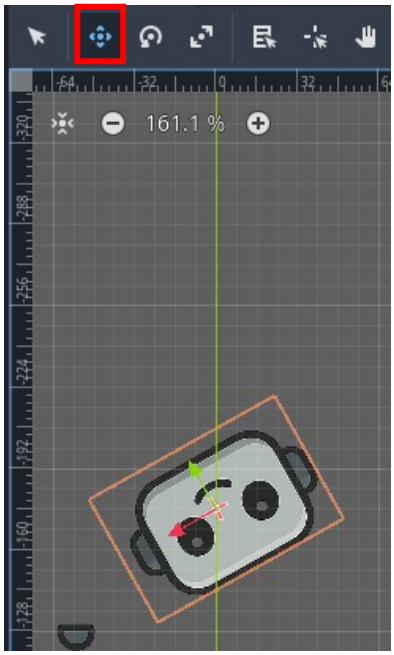
Don't forget to save your hard work! Press **CTRL + S** at the same time on the keyboard to save the project.

21 Use the **Tool Bar** to fix the Robot parts.

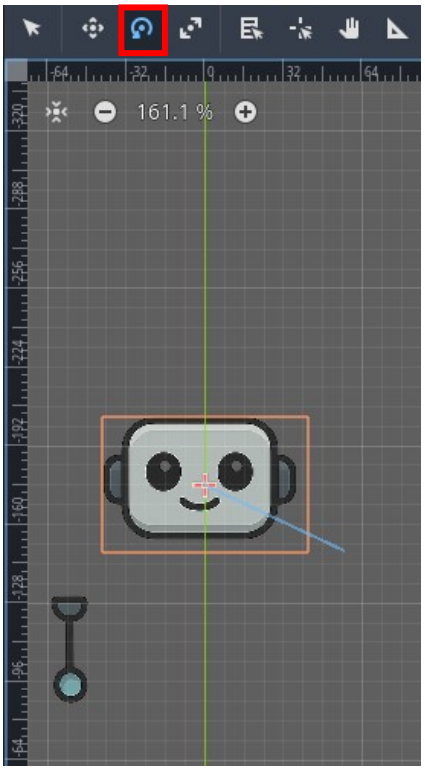
Start with the **Head** node. Use the **select** tool and click on the **Head** in the game window to select it.



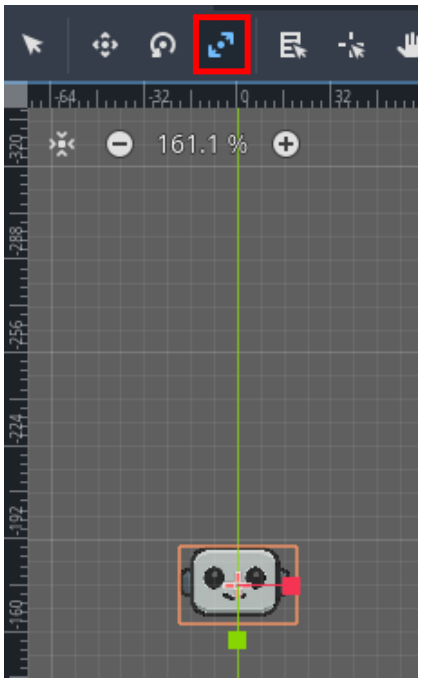
Use the **move** tool to move the **Head** node towards the center by dragging the node.



Use the **rotate**  tool to rotate the **Head** by holding left-click and moving the mouse at the same time.



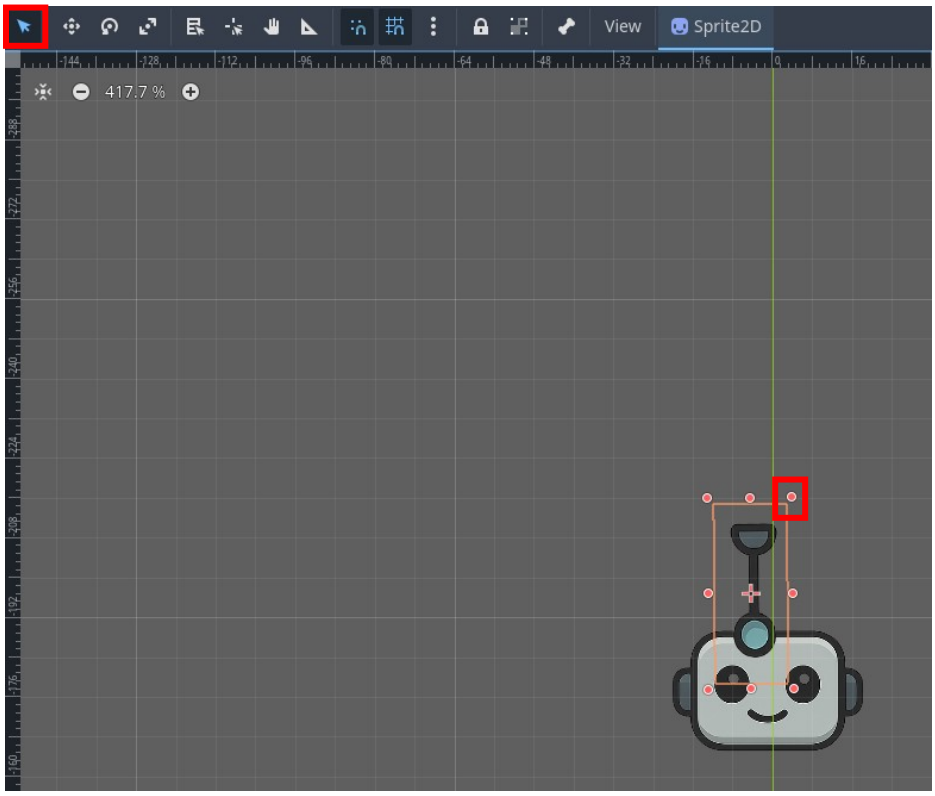
Use the **scale**  tool to resize the **Head** by dragging the **red arrow** to make it wider or narrower and the **green arrow** to make it shorter or longer.



Pro Tip:

Zoom in and out on the Godot editor to view all the Robot parts more easily.

22 To make fixing the Robot easier, use only the **select** and **rotate** tools to fix the **Antenna**. Place the **Antenna** in the center of the Robot's head with the circle at the top.

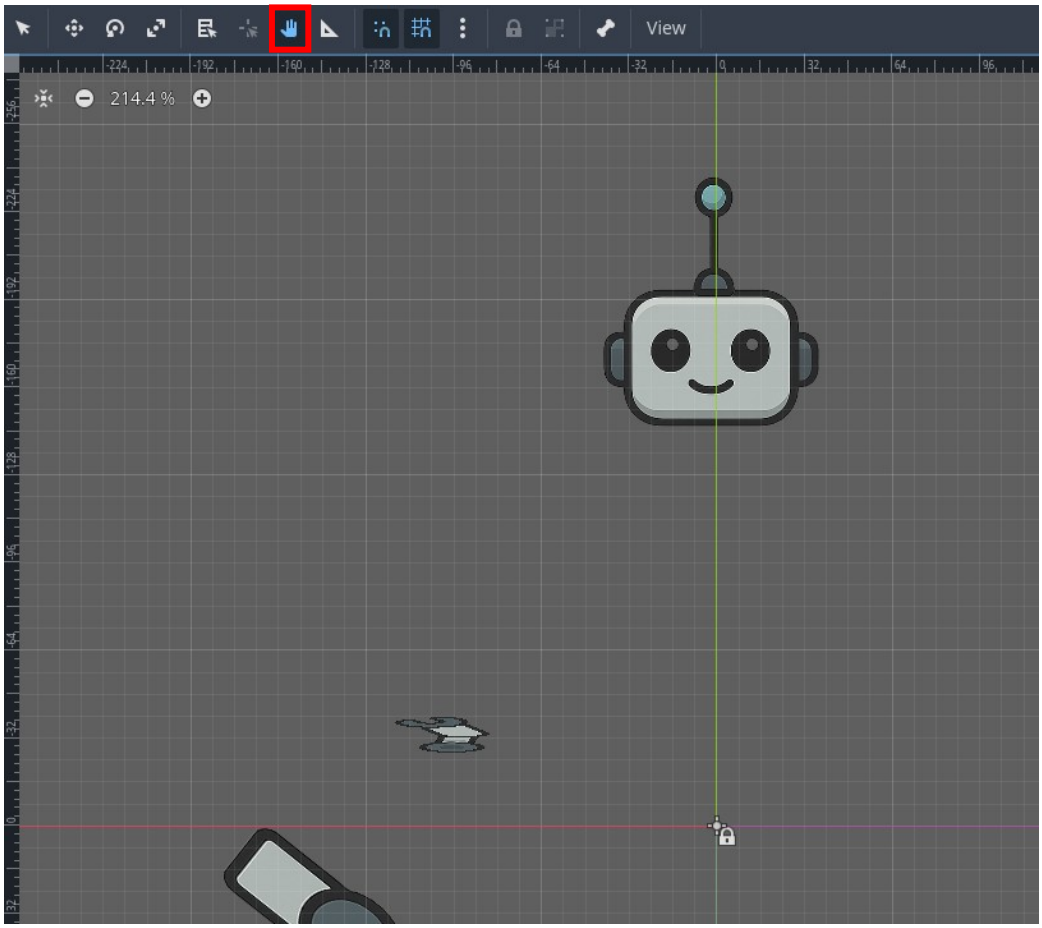


Pro Tip:

The **select** tool can be used to **move** and **resize** the nodes. Drag the nodes to change their position and drag the **red dot** in each corner to easily change its size.

23 Use the **Tool Bar** to fix the rest of the Robot parts.

Use the **pan** tool to click and drag the game window to navigate around the **Viewport** more easily.

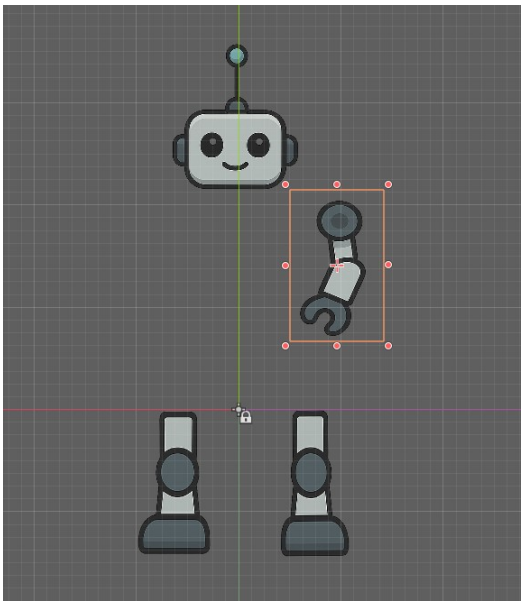


Reminder:

Switch back to the **select** tool to **move** and **resize** nodes. Only the **rotate** tool can rotate the node.

Refer to the **previous steps** for additional guidance!

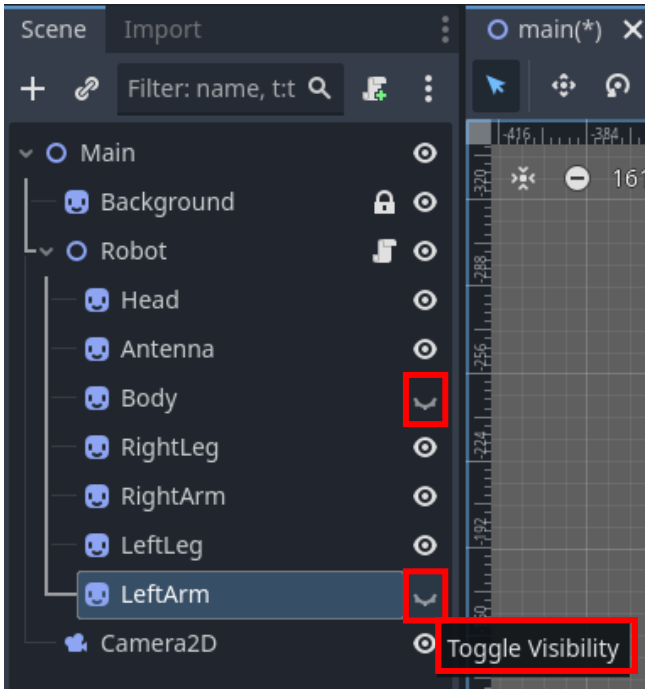
24 Hmmm some parts seem to be missing. Which parts are not visible?



25 Go to the **Scene Menu.**

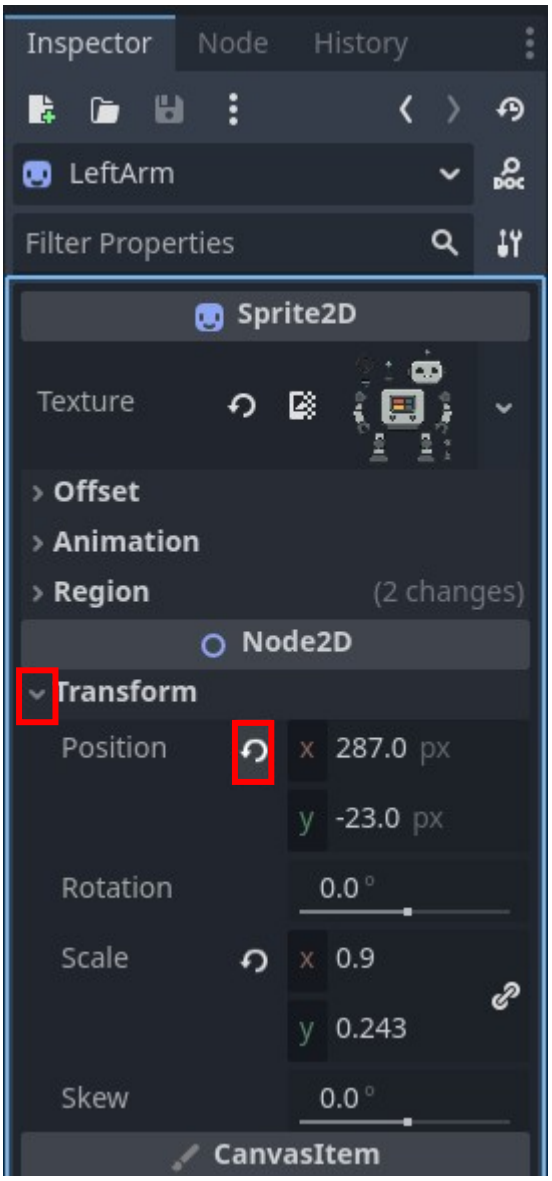
Notice that some nodes have closed eyes  . Hover over the symbol and notice the **Toggle Visibility** Text.

Click on the **closed eye symbols** to make the **Body** and **LeftArm** nodes visible.



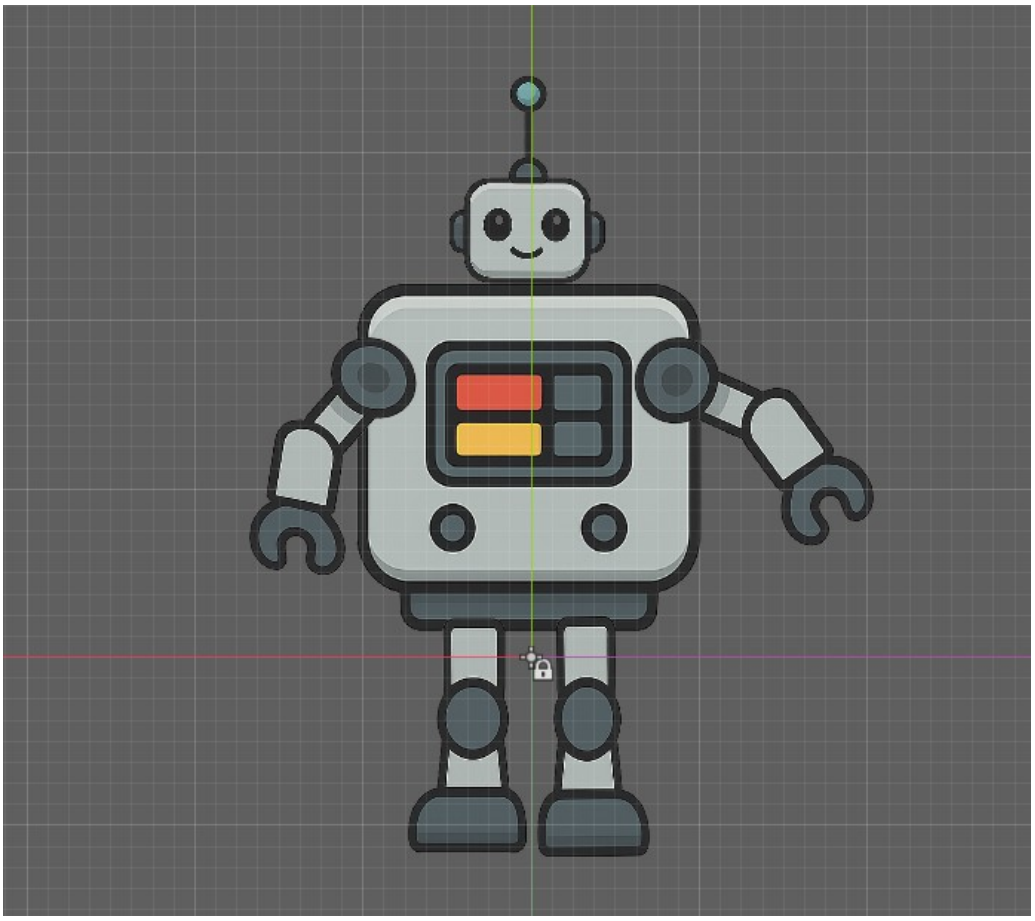
26 If a Robot part is not immediately visible, select the node in the **Scene Menu.**

In the **Inspector,** open the **Transform** drop-down menu and click on the **Reset Position** button. This will restore the part back to its origin.



27 Use the **Tools Bar** to fix the remaining Robot parts.

Feel free to adjust the size of each part differently to create a custom Robot!

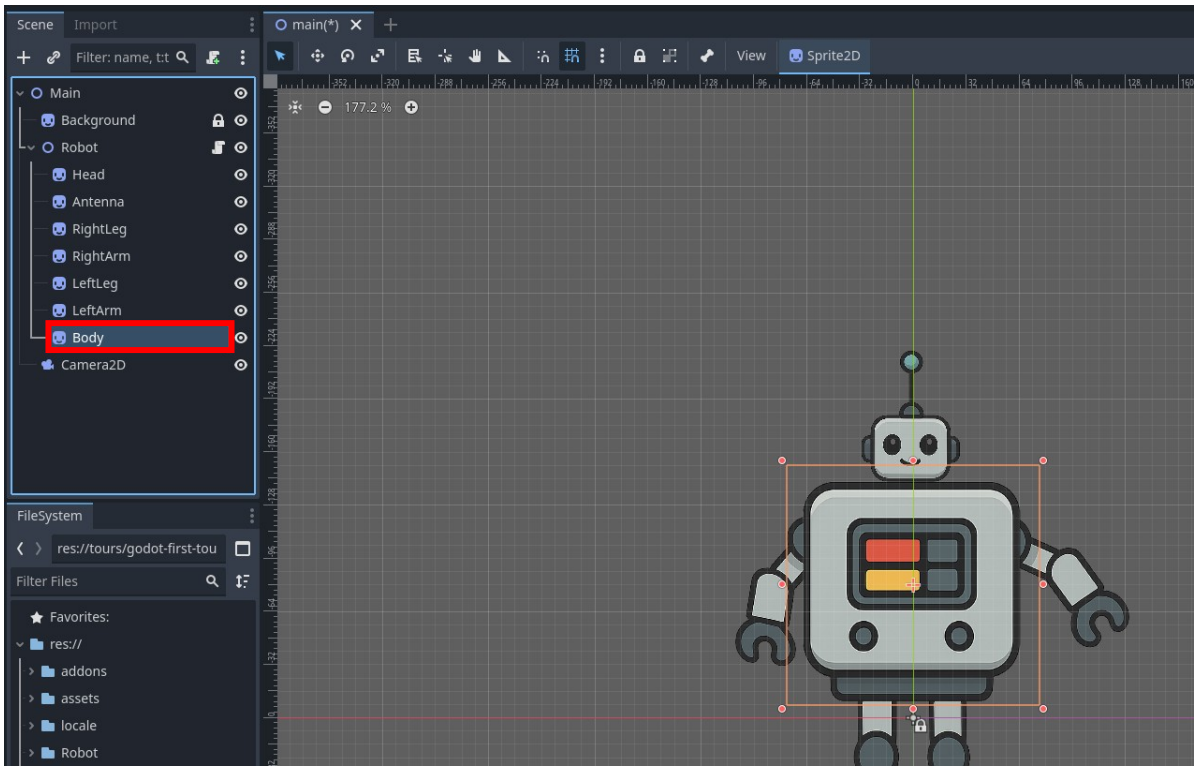


28 The Robot can also look different depending on the order of the Robot parts.

Nodes that are higher up in the **Scene Menu** will appear *behind* other nodes in the viewport.

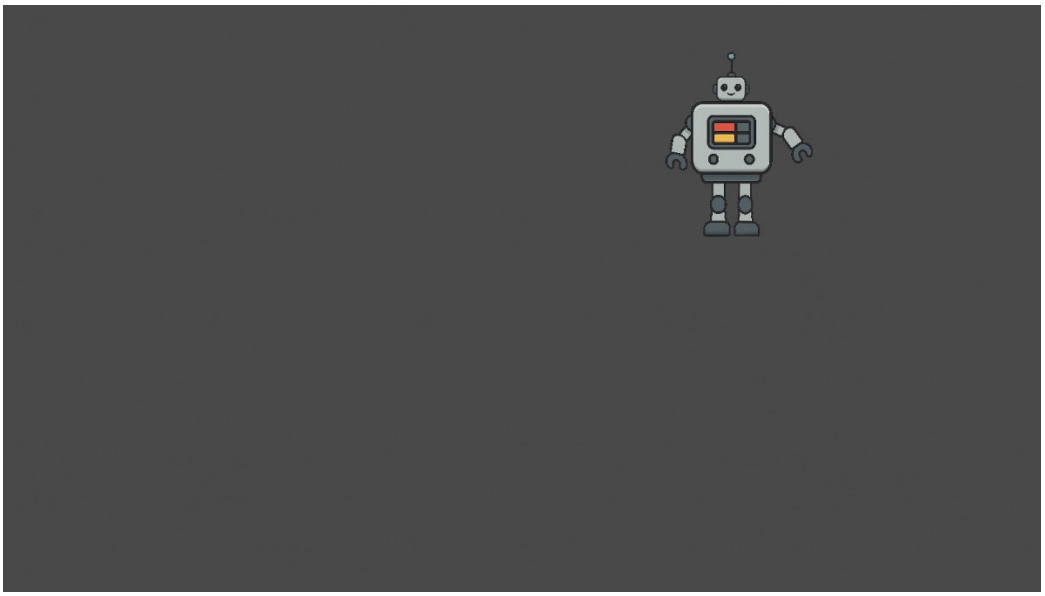
Rearrange the Robot parts by dragging and dropping them in a different order to see how that changes the Robot!

Notice the **Body** node is *below* other nodes in the **Scene Menu** and appears in *front* of the other parts.



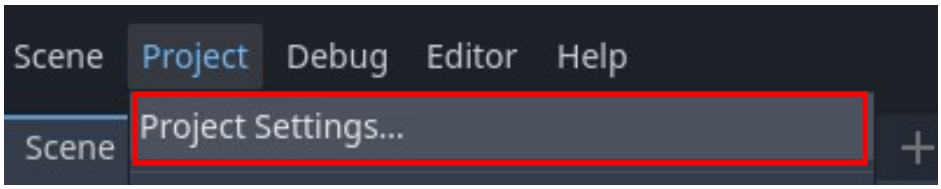
29 Click ▶ in the top right corner to playtest the project. Use the **directional buttons** to move the Robot.

Can the **Robot** be moved with the **direction buttons** *and* the **WASD keys** on the keyboard?

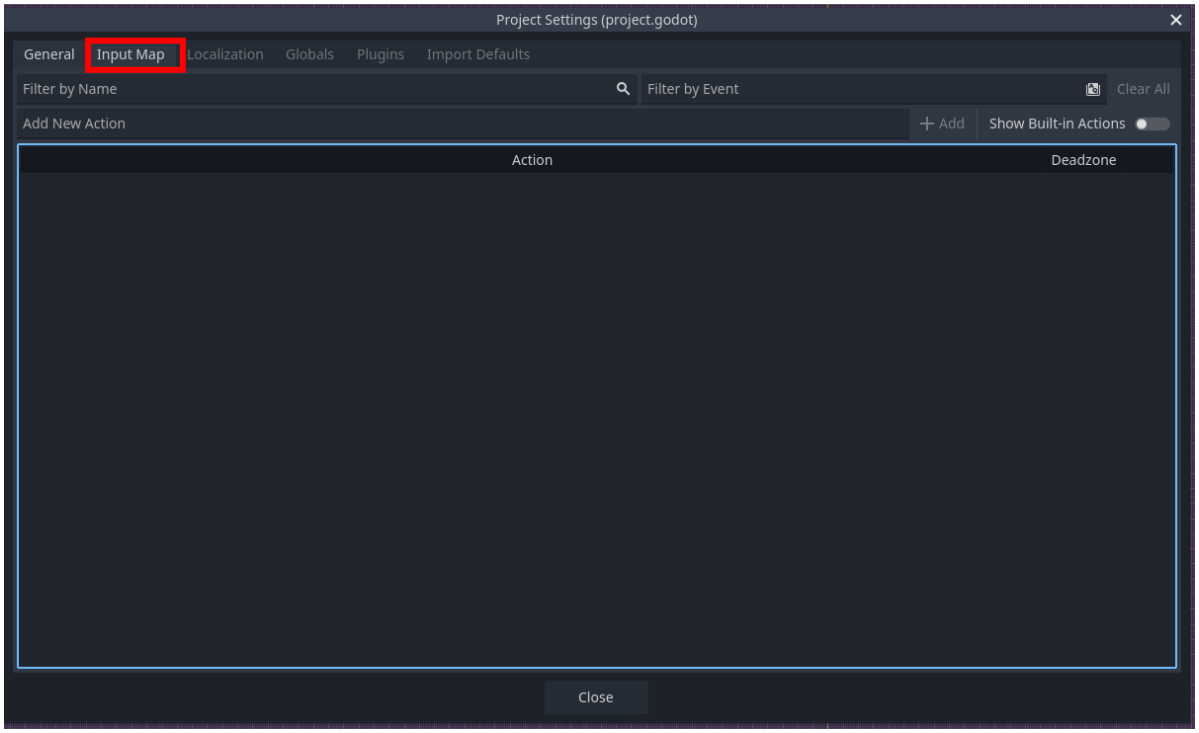


30 Currently, pressing the WASD keys do not move the Robot.

In the top left corner of the editor, click on **Project** then select **Project Settings** from the drop-down menu.



In the pop-up window, click on **Input Map** at the top. Godot's **Input Map** allows **inputs** to be defined and set to different keys or buttons.



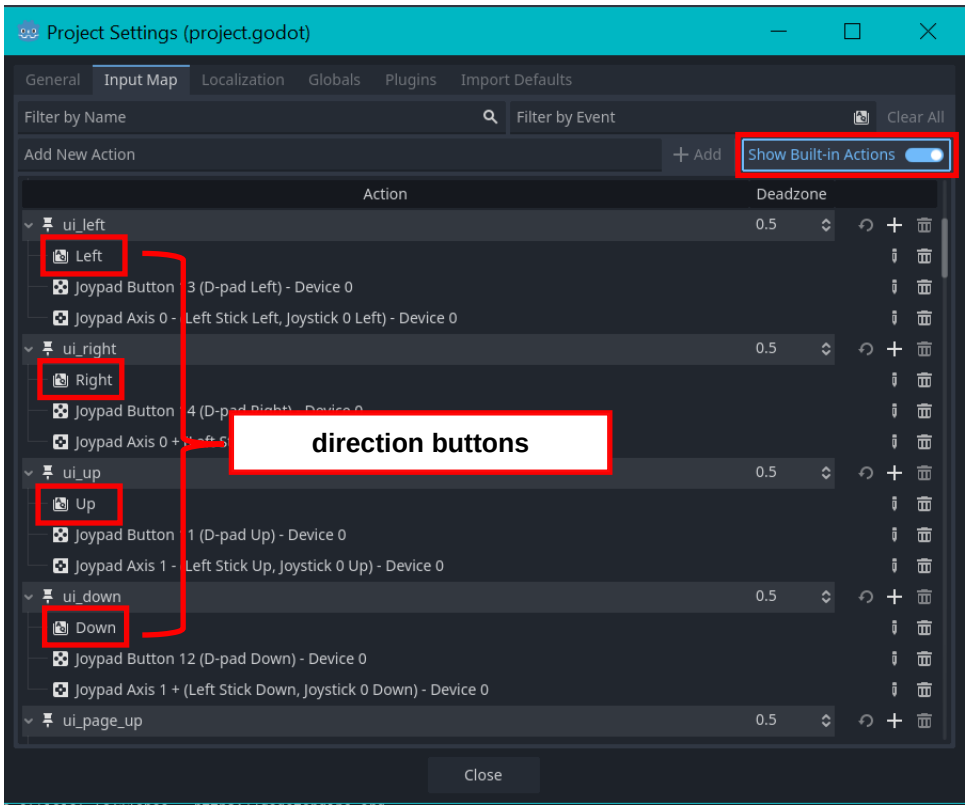
Reminder:

Inputs are another word for what the computer and game receive from the user!

31 What Inputs is the game currently receiving?

In the top right corner, click on **Show Built-In Actions**. This will display all of Godot's built in actions such as moving left, right, up, and down.

Scroll down to view **ui_left**, **ui_right**, **ui_up**, and **ui_down**. These are variables for when the Robot moves left, right, up, or down.



Currently, the only inputs on the keyboard that are set to move the Robot are the **direction buttons**.

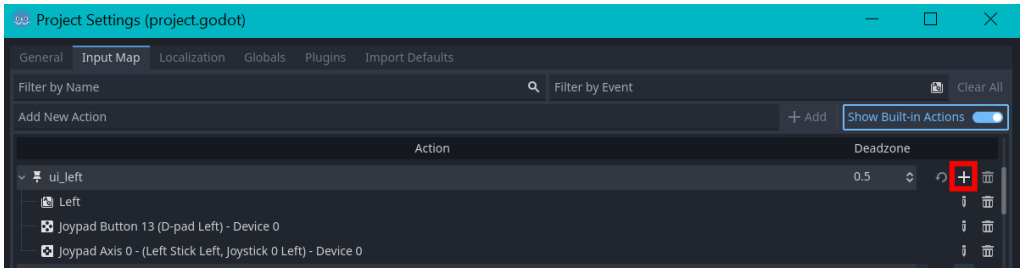
Pro Tip:

Don't worry about the **Joypad** inputs; these would be used to program a controller.

32

Add the **WASD** keys as an input to move the Robot.

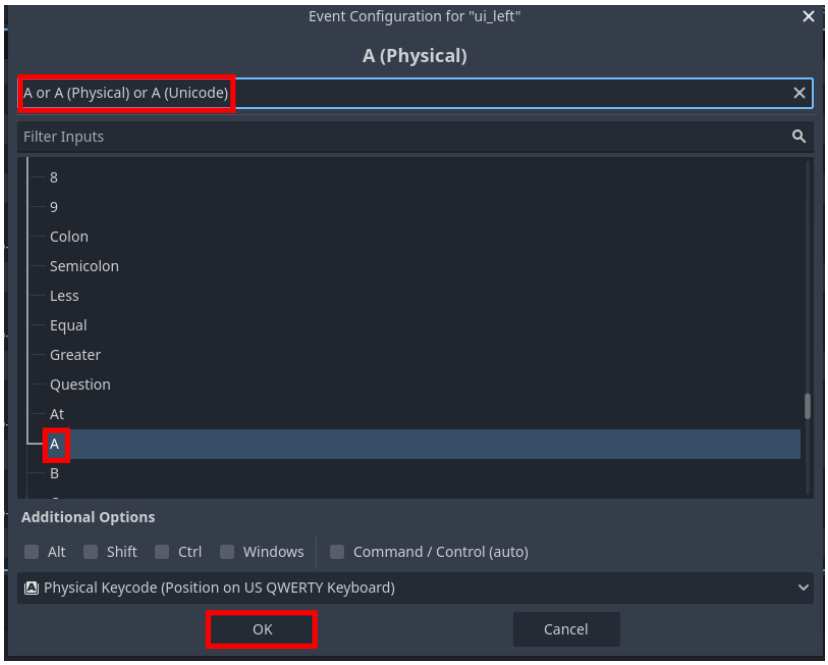
Click on the **+** symbol to the right of **ui_left**.



Since **ui_left** is the variable for when the Robot moves to the left, set the **A** key to it. Type **A** in the search bar.

Press **only** the **A key**. Do **not** also press down on **shift** to capitalize or else **A + shift** will be added as an input and not **A** by itself.

Find the **A** key, then click **OK**. Now, the Robot will move to the left when either the left direction button or A key is pressed.



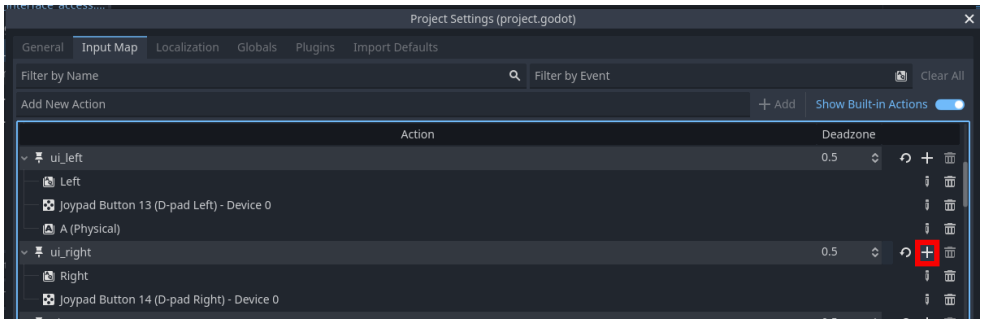
Pro Tip:

Make sure **A (Physical)** was added and not a different key!

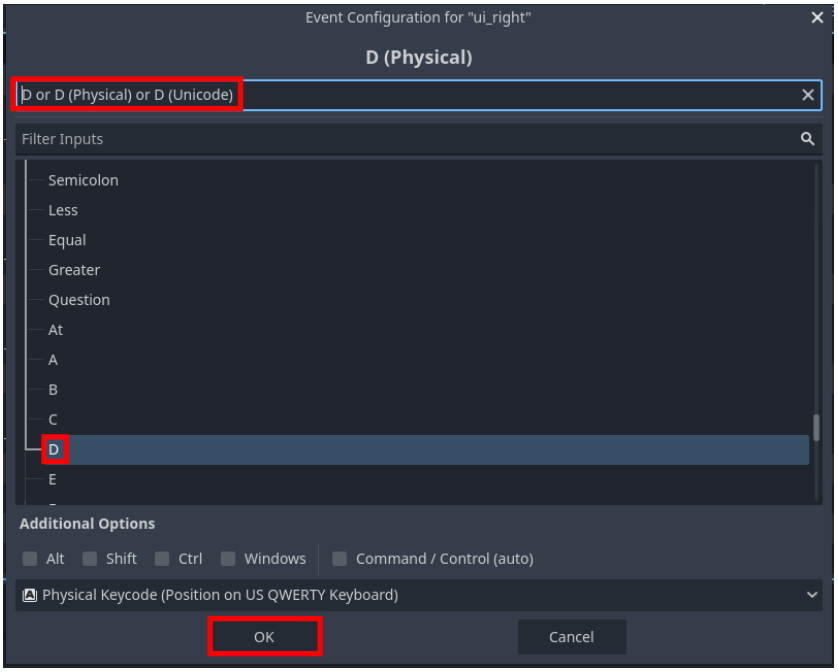
33

Add the **D** key as an input to move the Robot to the right.

Click on the **+** symbol to the right of **ui_right**. **Ui_right** is the variable for when the Robot moves to the right.



Type only **D** in the search bar, and make sure that **D + shift** is not the input selected.



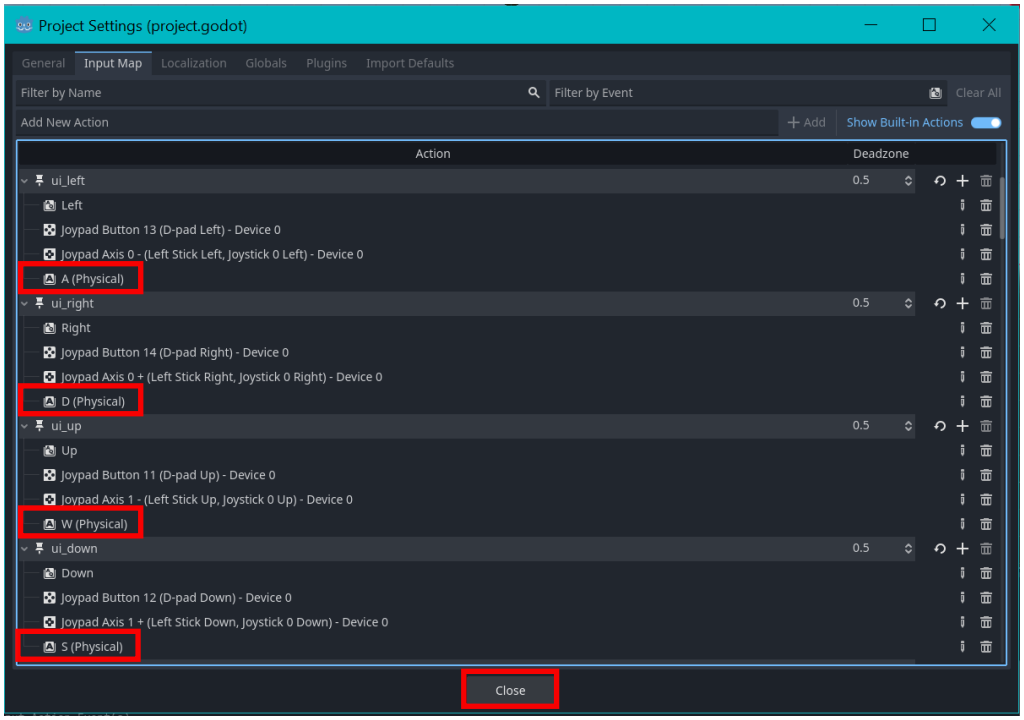
Find the **D** key, then click **OK**. Now, the Robot will move to the right when either the right direction button or D key is pressed.

Pro Tip:

Make sure **D (Physical)** was added and not a different key!

34 Repeat the previous steps for the **ui_up** and **ui_down** variables so that all **WASD** keys are added.

Click **Close** once all keys have been added.



Pro Tip:

When adding the keys **only** type the letter on the keyboard into the search, do **not** press **shift!**

35 Click ▶ in the top right corner to playtest the project again. Check and make sure the **WASD** keys now move the Robot.

Pause for **Sensei Stop #5!**

Ask a Code Sensei to playtest the final project! Discuss with your Code Sensei:

- What tool or feature in Godot did you find confusing, and what would help you understand it better?
- What do you want to create next in Godot, using the tools you've learned?

Reminder: Press **CTRL + S** on the keyboard to save the project.

Congratulations on completing **BB Activity 00: Welcome to Godot – You Rock!**

Continue your exploration with Godot by opening the **BB Activity 01: Dropping Bombs** Ninja Guide.