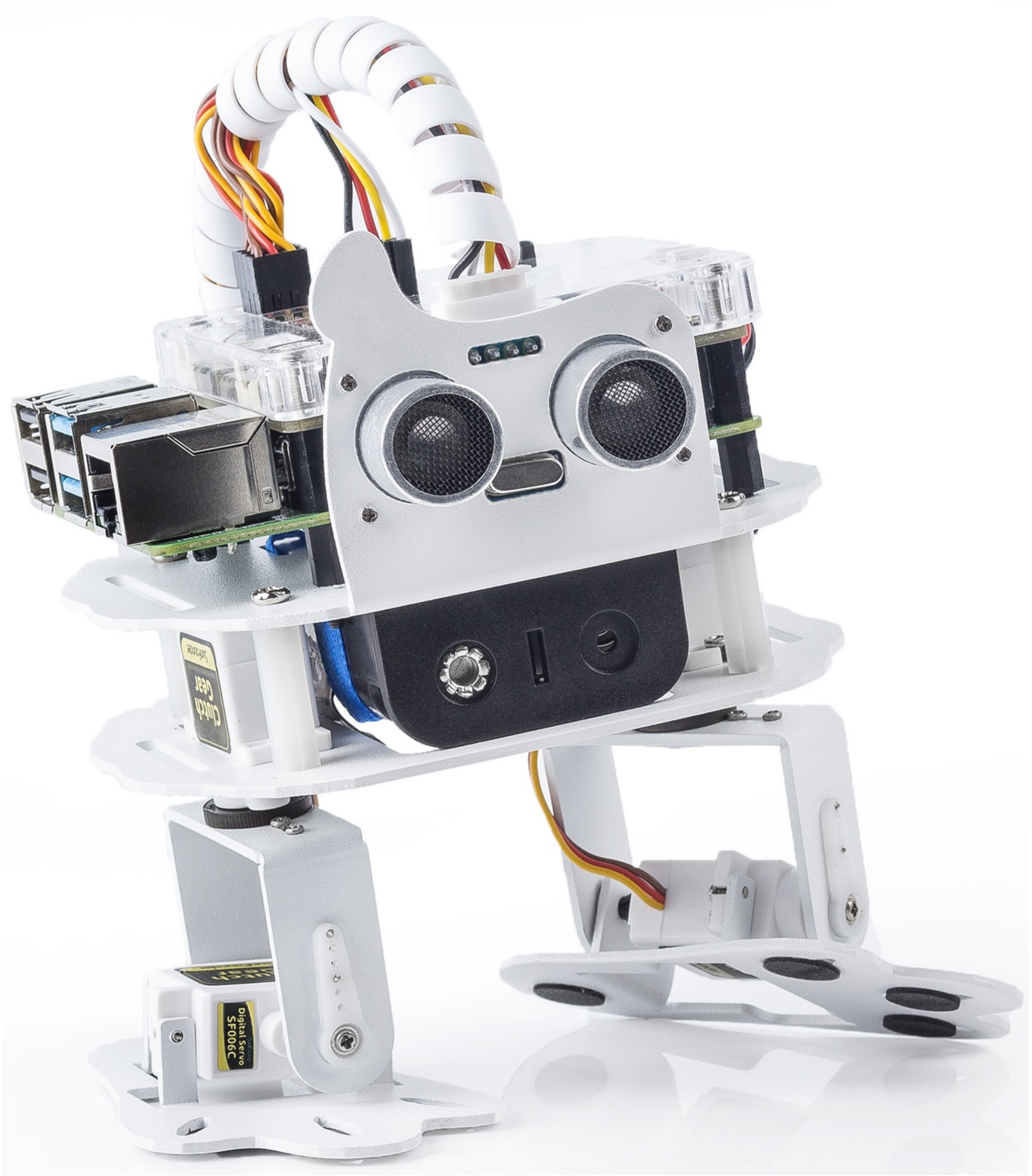
SunFounder pisloth

www.sunfounder.com

Aug 02, 2022

CONTENTS

Thanks for choosing our PiSloth.



PiSloth is a Raspberry Pi Bionic robot with an aluminum alloy structure. It can talk, dance, and even express emotions, such as happiness and excitement.

It has 22 different actions, such as: Stomp, Swing and MoonWalk, and you can customize the actions according to your needs. PiSloth's eyes consist of an ultrasonic sensor module that can be used to detect distance for obstacle avoidance and following functions.

In this tutorial, a list and assembly pdf, introduction to Robot HAT, and programming of PiSloth are included.

The programming part is divided into two chapters: *Play with Ezblock* & *Play with Python*, and each of them can get you stated on making PiSloth work in way you want.

EzBlock Studio is a development platform developed by SunFounder designed for beginners to lower the barriers to getting started with Raspberry Pi. It has two programming languages: Graphical and Python, and available on almost all different types of devices. With Bluetooth and Wi-Fi support, you can download code, remote control a Raspberry Pi, on Ezblock Studio.

More experienced makers can use the popular programming language - Python.

Content

CHAPTER

ONE

COMPONENT LIST AND ASSEMBLY INSTRUCTIONS

You need to check whether there are missing or damaged components according to the list first. If there are any problems, please contact us and we will solve them as soon as possible.

Please follow the steps on the PDF to assemble.

Note:

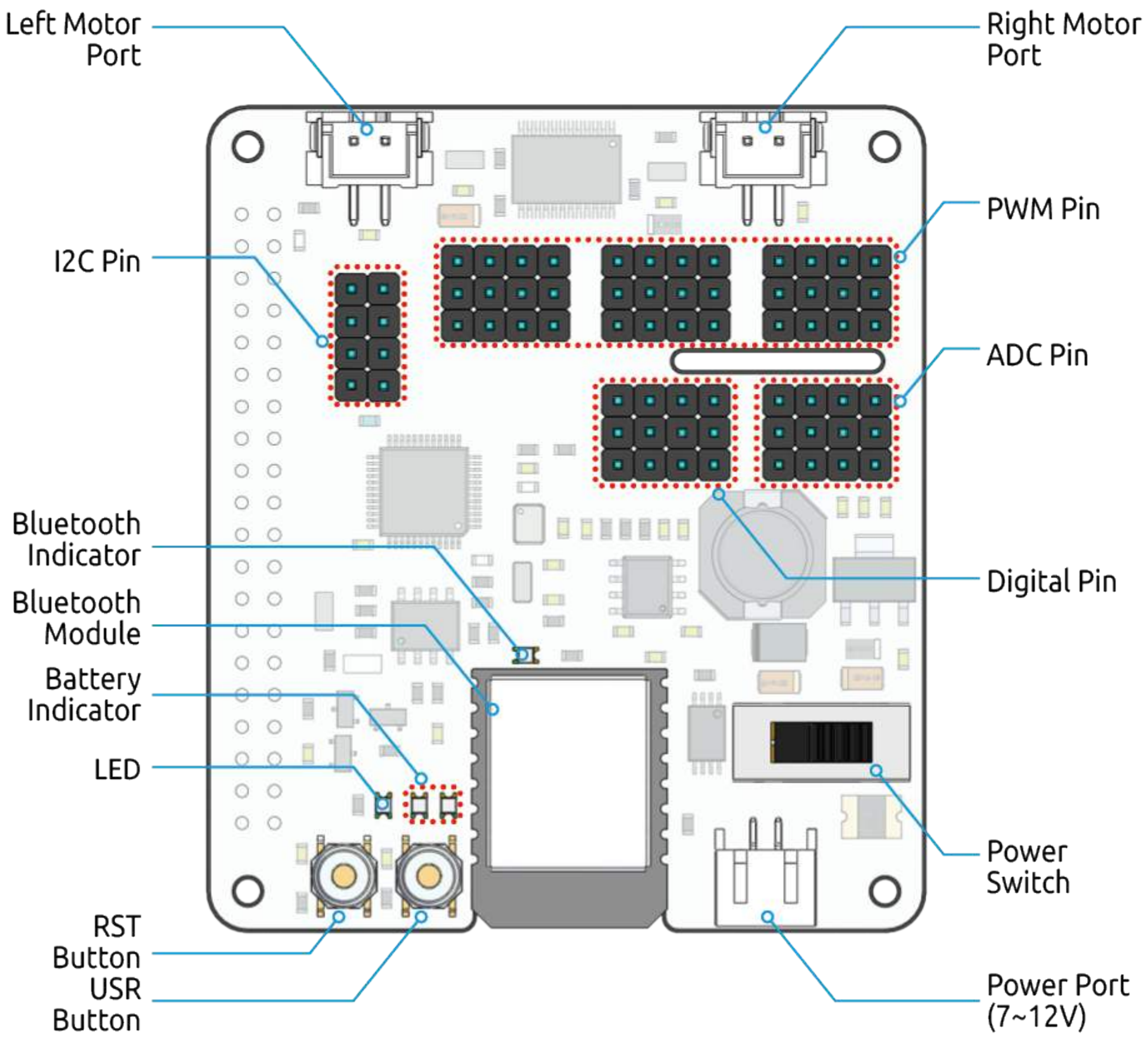
1. Before assembling, you need to buy 2 18650 batteries and fully charge them, refer to *About the Battery*.
2. Robot HAT cannot charge the battery, so you need to buy a battery charger at the same time.

- `Component List and Assembly Instructions`.

CHAPTER

TWO

ABOUT ROBOT HAT



Robot HAT is a multifunctional expansion board that allows Raspberry Pi to be quickly turned into a robot. An MCU is on board to extend the PWM output and ADC input for the Raspberry Pi, as well as a motor driver chip, Bluetooth module, I2S audio module and mono speaker. As well as the GPIOs that lead out of the Raspberry Pi itself.

It also comes with a Speaker, which can be used to play background music, sound effects and implement TTS functions to make your project more interesting.

Accepts 7-12V PH2.0 2pin power input with 2 power indicators. The board also has a user available LED and a button for you to quickly test some effects.

Note: You can see more details in the Robot HAT Documentation.

CHAPTER

THREE

PLAY WITH EZBLOCK

For beginners and novices, **EzBlock** is a software development platform offered by SunFounder for Raspberry Pi. Ezbock offers two programming environments: a graphical environment and a Python environment.

It is available for almost all types of devices, including Mac, PC, and Android.

Here is a tutorial to help you complete EzBlock installation, download, and use.

3.1 Quick Guide on EzBlock

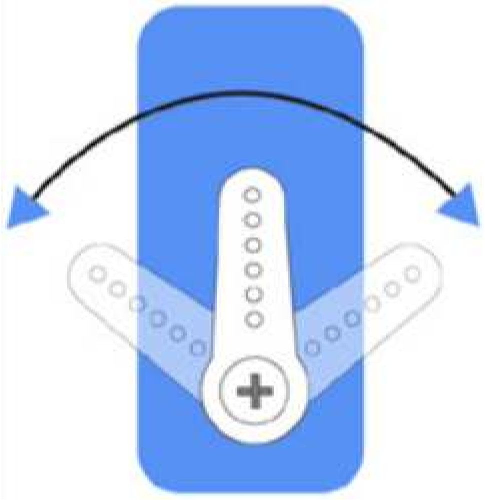
There are 2 parts here:

- *Servo Adjust* allows you to keep all the servos at 0 degrees to complete a proper and safe assembly (otherwise you will probably damage the servos).
- *Install and Configure EzBlock Studio* will guide you to download EzBlock Studio to play with your robot.

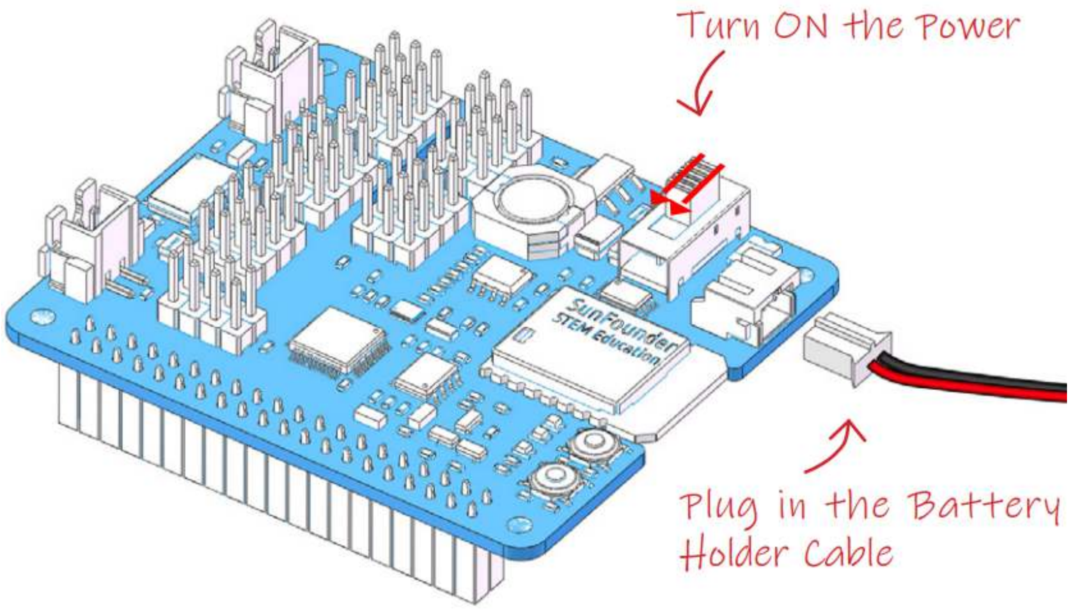
3.1.1 Servo Adjust

When assembling to the part with the servo, you need to keep the servo at 0° and secure it with the servo screw. Please follow the tutorial below to do this.

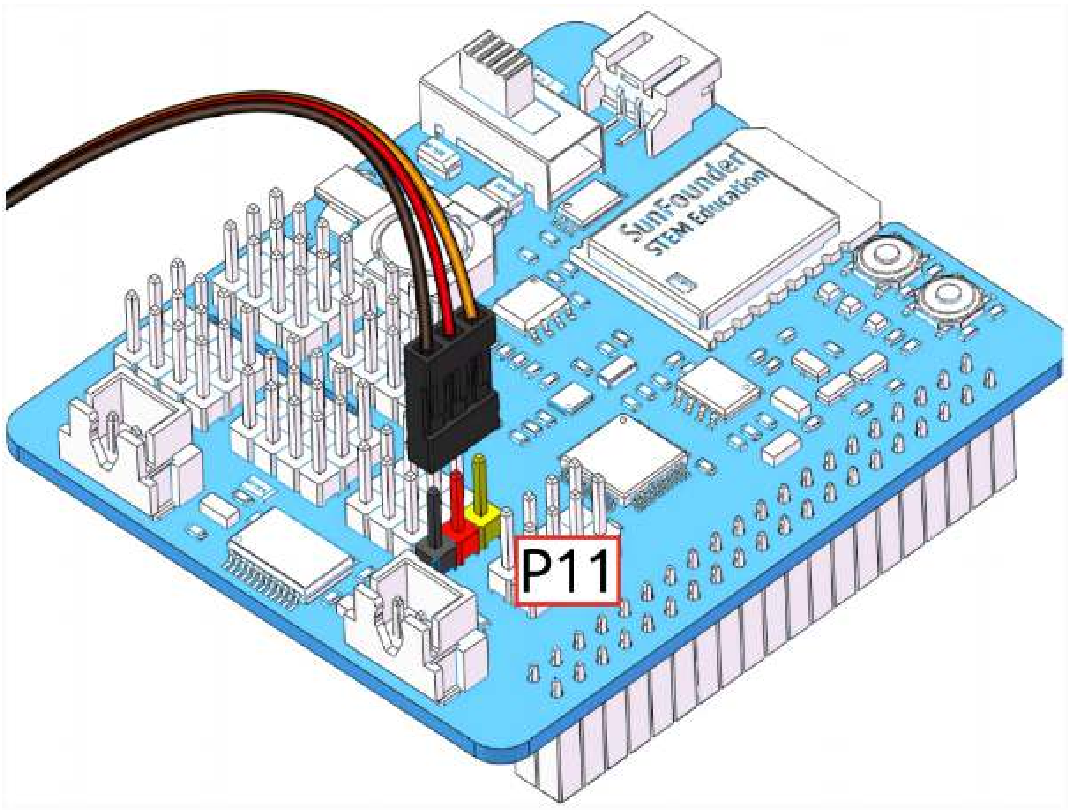
1. Firstly, Install EzBlock OS(3.1) onto a Micro SD card, once the installation is complete, insert it into the Raspberry Pi.
2. To ensure that the servo has been properly set to 0°, first insert the rocker arm into the servo shaft and then gently rotate the rocker arm to a different angle.



3. Follow the instructions on the assembly foldout, insert the battery holder cable and turn the power switch to the ON. Wait for 1-2 minutes, there will be a sound to indicate that the Raspberry Pi boots successfully.



4. Next, plug the servo cable into the P11 port as follows.



5. At this point you will see the servo arm rotate to a specific position (0°). If the servo arm does not return to 0°, press the RST button to restart the Robot HAT.
6. Now you can continue the installation as instructed on the assembly foldout.

Note:

- Do not unplug this servo cable before fastening this servo with the servo screw, you can unplug it after fastening.
- Do not turn the servo while it is powered on to avoid damage; if the servo shaft is inserted at the wrong angle, pull out the servo and reinsert it.
- Before assembling each servo, you need to plug the servo cable into P11 and turn on the power to set its angle to 0°.
- This zeroing function will be disabled if you download a program to the robot later with the EzBlock APP.

3.1.2 Install and Configure EzBlock Studio

As soon as the robot is assembled, you will need to carry out some basic operations.

- Install EzBlock Studio(3.1): Download and install EzBlock Studio on your device or use the web-based version.
- Connect the Product and EzBlock(3.1): Configure Wi-Fi, Bluetooth and calibrate before use.
- Open and Run Examples(3.1): View or run the related example directly.

Note: After you connect the PiSloth, there will be a calibration step. This is because of possible deviations in the installation process or limitations of the servos themselves, making some servo angles slightly tilted, so you can calibrate them in this step.

But if you think the assembly is perfect and no calibration is needed, you can also skip this step.

Projects

Here, we show you the projects of playing PiSloth on EzBlock Studio. If you are new to these, you can refer to the code images inside each project to program, and can learn the use of blocks according to TIPS.

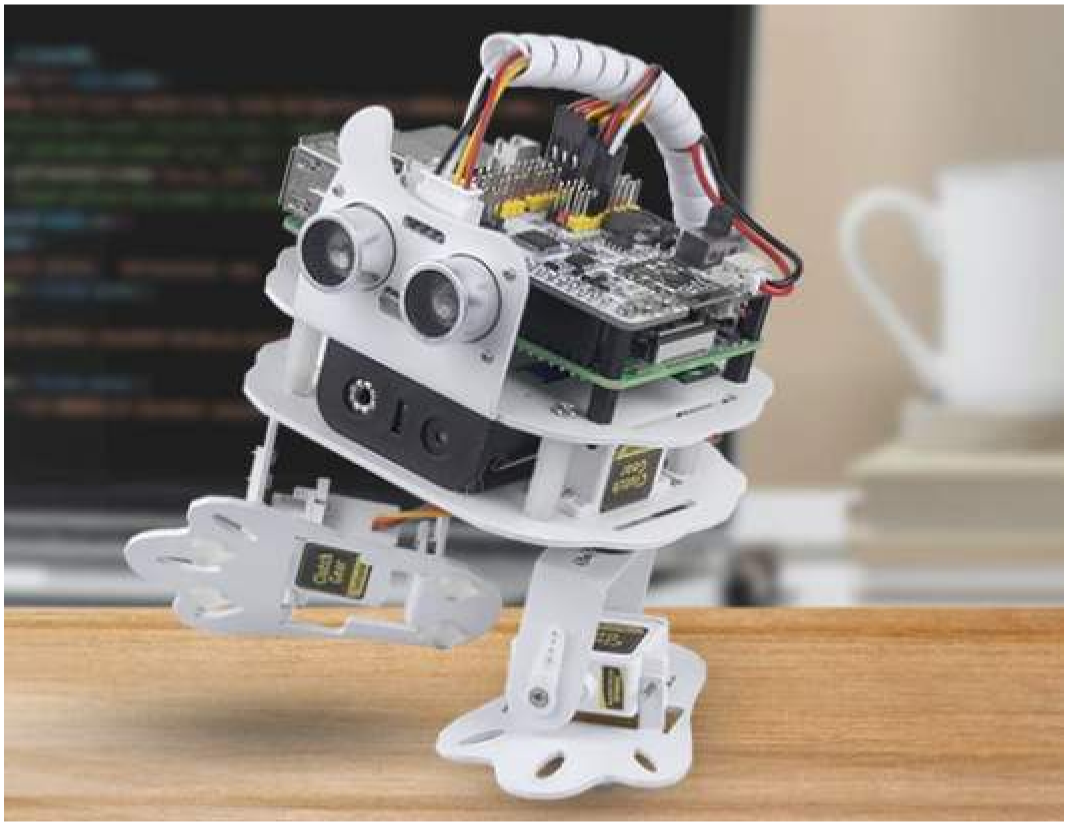
If you don't want to write these projects one by one, we have uploaded them to EzBlock Studio's Examples page and you can run them directly or edit them and run them later.

3.2 Move

This is the first project. PiSloth has woken up, and it moves freely.

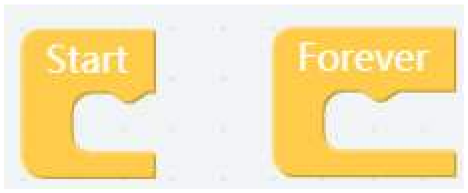
Before programming, you need to learn the basic usage of EzBlock Studio from here.

- How to Create a New Project?



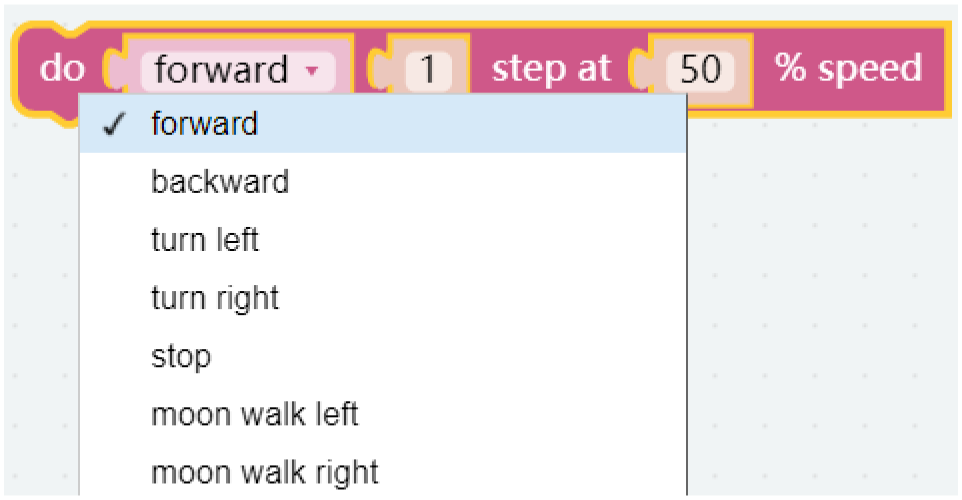
TIPS

This is the basic structure of the program, the Start block is used to do some initialization (even if no block is placed, it cannot be deleted) and the Forever block is, as the name suggests, a continuous loop that allows your program to change and respond.

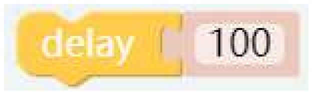


This block is used to make PiSloth do a specific action several steps at a speed (%), for example, let PiSloth go forward 1 step at 50% speed.

Different actions can be selected from the drop down options, there are 22 in total.



This is a block that sets the duration of the previous block, unit: ms.



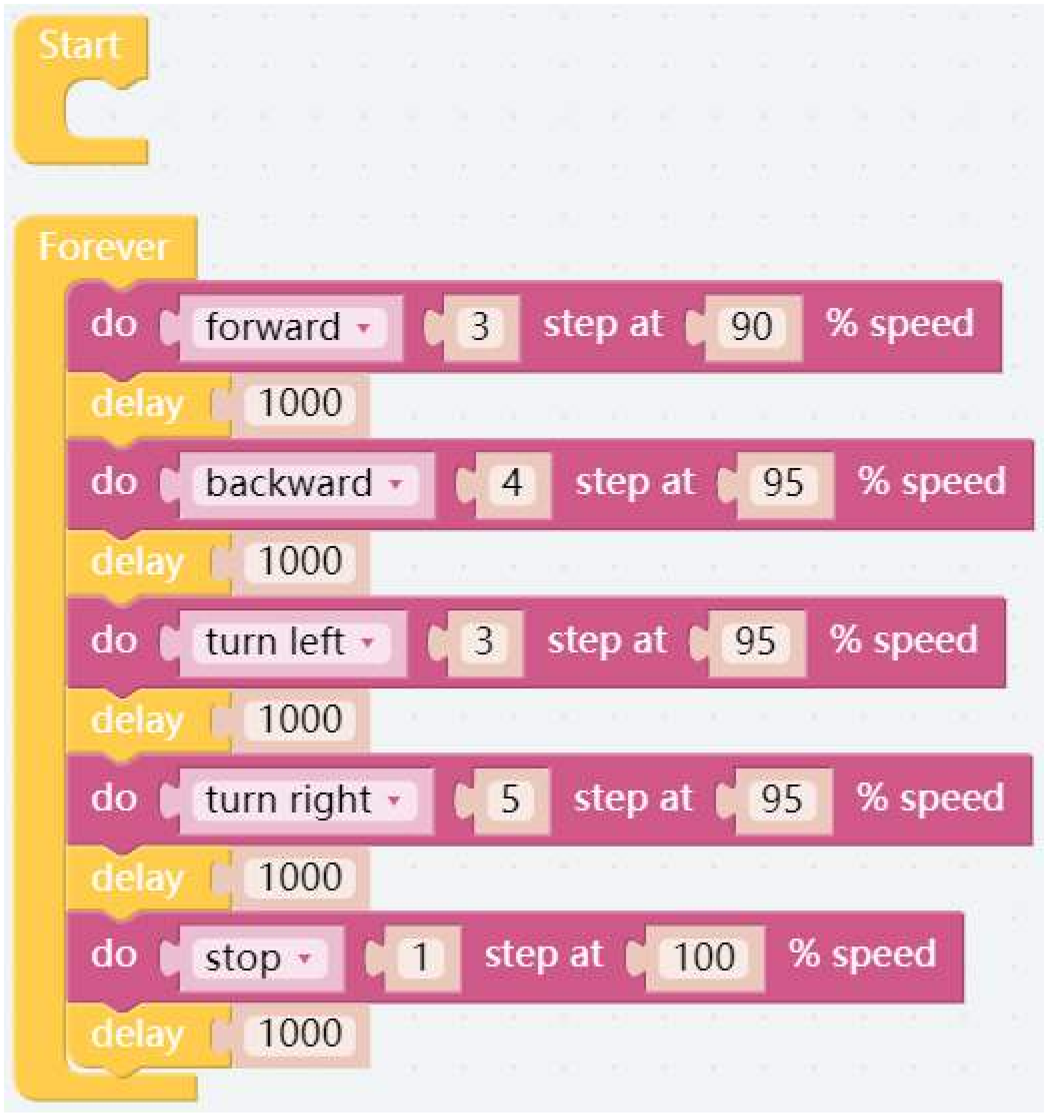
EXAMPLE

Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.

After writing the code according to the following figure, click the download icon in the bottom right corner, you will see PiSloth move forward 3 steps, backward 4 steps, left 3 steps, right 5 steps, and finally stop. Since the whole code is placed inside the Forever block, PiSloth will repeat the above actions after stopping for a while.

You can try putting the code from the Forever block into the Start block and see what happens.



```
Start

Forever
  do forward 3 step at 90 % speed
  delay 1000
  do backward 4 step at 95 % speed
  delay 1000
  do turn left 3 step at 95 % speed
  delay 1000
  do turn right 5 step at 95 % speed
  delay 1000
  do stop 1 step at 100 % speed
  delay 1000
```

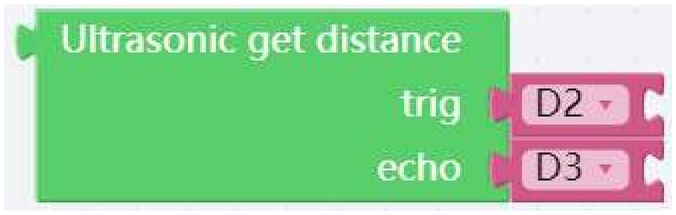
3.3 Don't Touch Me

If you don't meet PiSloth's needs, it will get angry and stay away from your touch.

TIPS

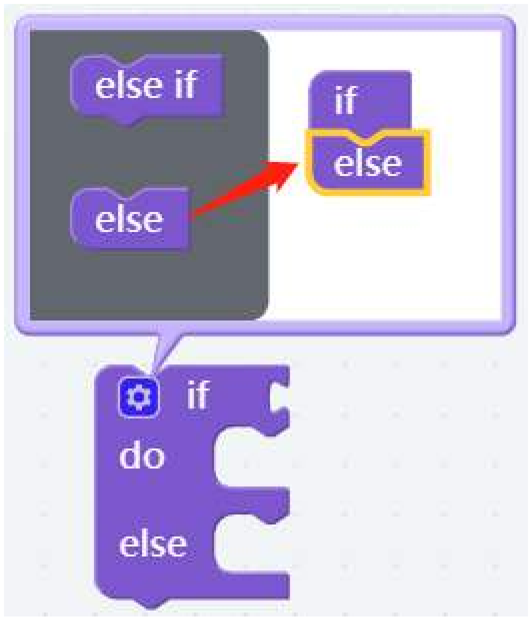
You can directly use this block to read the distance to the obstacle right ahead.

Note: When assembling, Trig and Echo are connected to D2 and D3 respectively, you also need to change them simultaneously when programming.

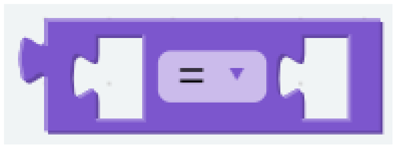


To achieve conditional judgment of “if” type, you need to use an **if do** block.

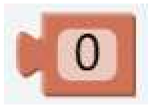
When you need to implement multiple conditional judgments, you will have to change **if do** into **if else do**. This can be achieved by clicking on the **setting** icon.



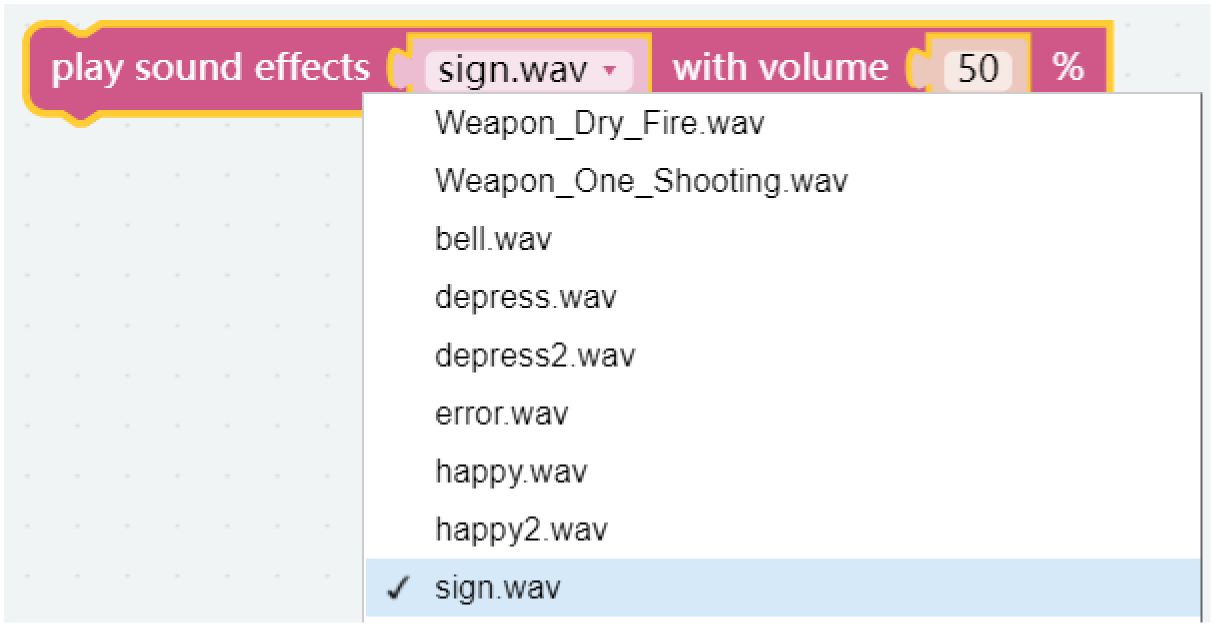
You need to use a conditional statements block in conjunction with if do. Judging conditions can be “=”, “>”, “<”, “”, “”, “”.



A number block.



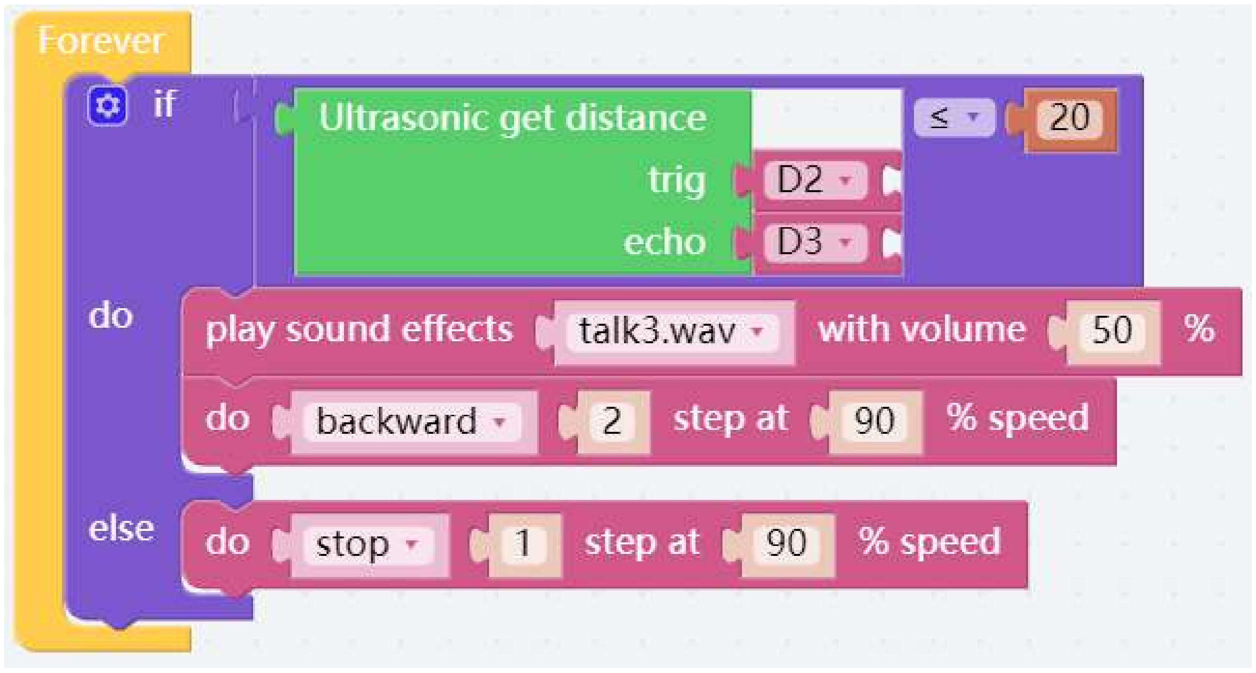
This block can emit some preset sound effects, such as siren sound, gun sound and so on. The range of volume is 1~100.



EXAMPLE

Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.


Forever
if
Ultrasonic get distance
trig D2
echo D3
≤ 20
do
play sound effects talk3.wav with volume 50 %
do backward 2 step at 90 % speed
else
do stop 1 step at 90 % speed

3.4 Obstacle Avoidance

In this project, when PiSloth detects an obstacle, it will send a signal and look for another direction to move forward.

TIPS

This is based on the previous project *Don't Touch Me*, which adds autonomous judgment, so that PiSloth can actively avoid obstacles in front of it.

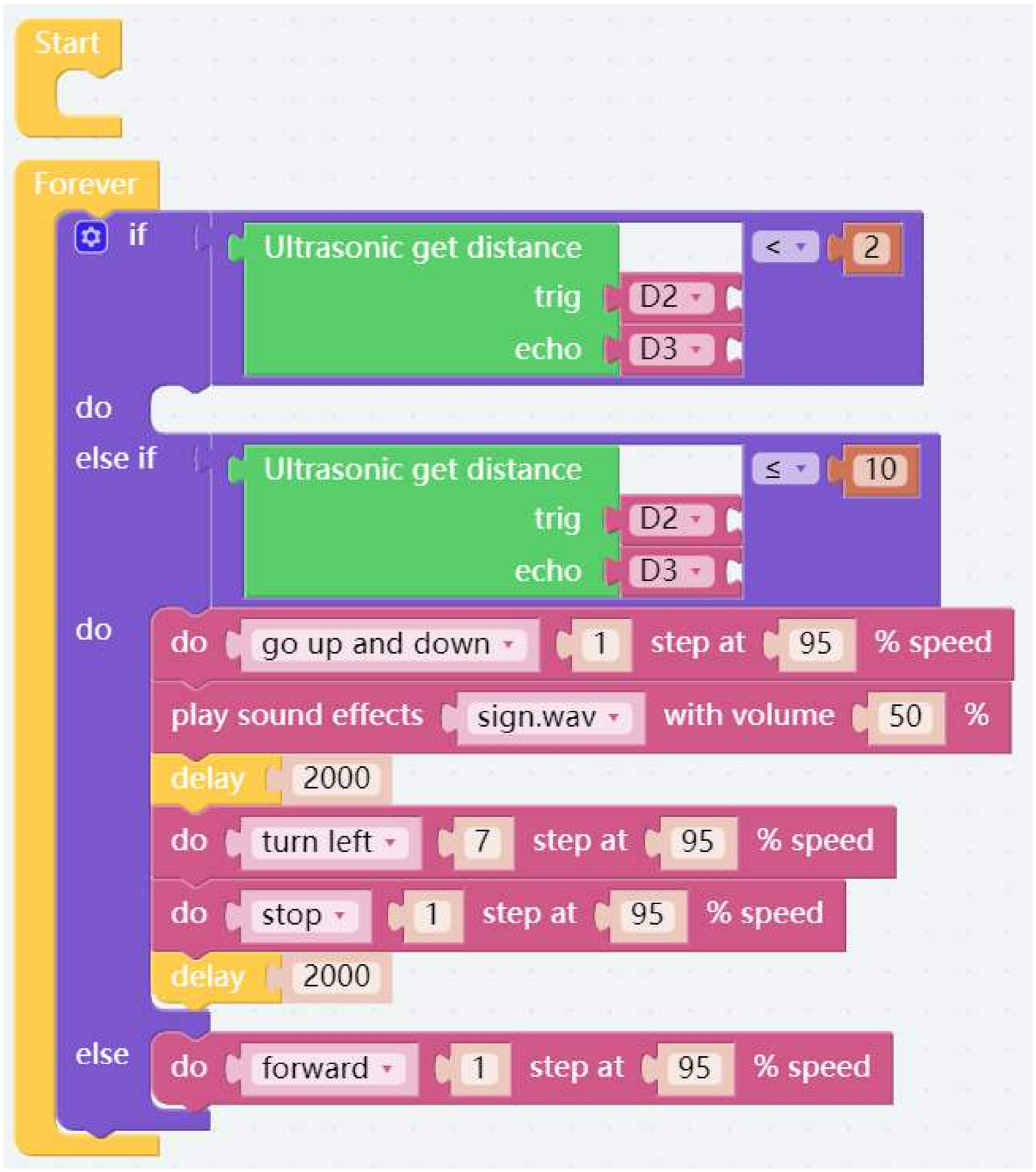
EXAMPLE

Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.

After the code runs, PiSloth will walk forward. If it detects that the distance of the obstacle ahead is less than 10cm, it will stop and sound a warning, then turn left for 7 steps and stop. If there is no obstacle in the direction after turning left or the obstacle distance is greater than 10, it will continue to move forward.

Since the effective detection distance of the ultrasonic sensor module is 2-400cm, when the detection distance is less than it will do nothing.



```
Start

Forever
  if  Ultrasonic get distance trig D2 echo D3  <  2
  do
  else if  Ultrasonic get distance trig D2 echo D3  ≤  10
  do
    do go up and down  1  step at  95  % speed
    play sound effects  sign.wav  with volume  50  %
    delay  2000
    do turn left  7  step at  95  % speed
    do stop  1  step at  95  % speed
    delay  2000
  else
    do forward  1  step at  95  % speed
```

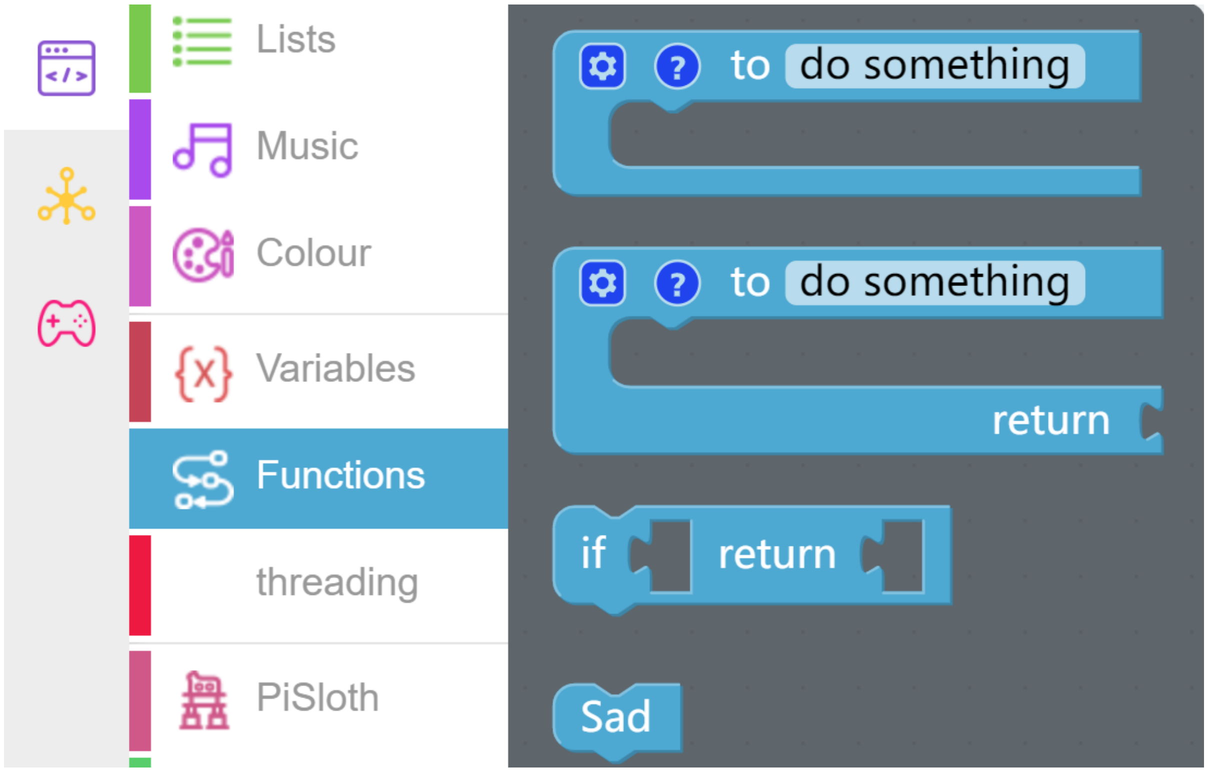
3.5 Emotional PiSloth

PiSloth is very emotional, sometimes happy, sometimes shy, sometimes confused.

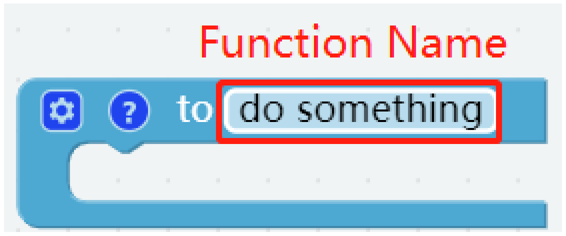
TIPS

You may want to simplify the program with **Functions**, especially when you perform the same operation multiple times. Putting these operations into a newly declared function can greatly facilitate your use.

Click on the **Functions** category and select the appropriate function block, the function you created will also appear here.



The **Function** block without output is used here.

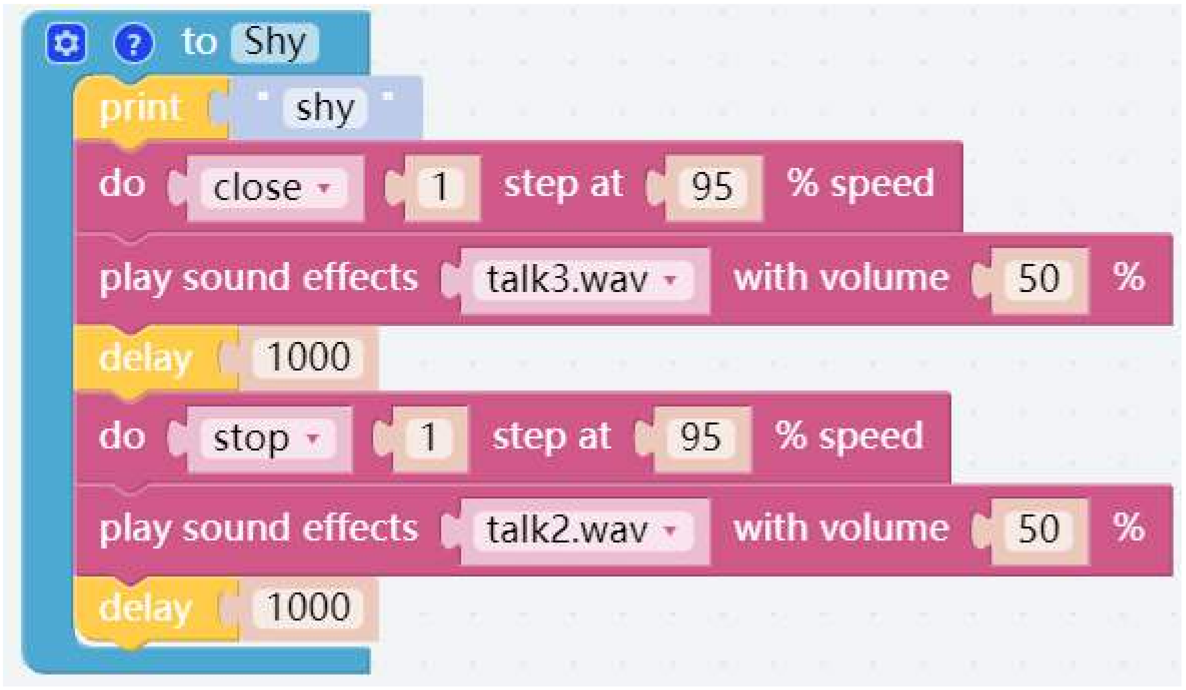


EXAMPLE

Note:

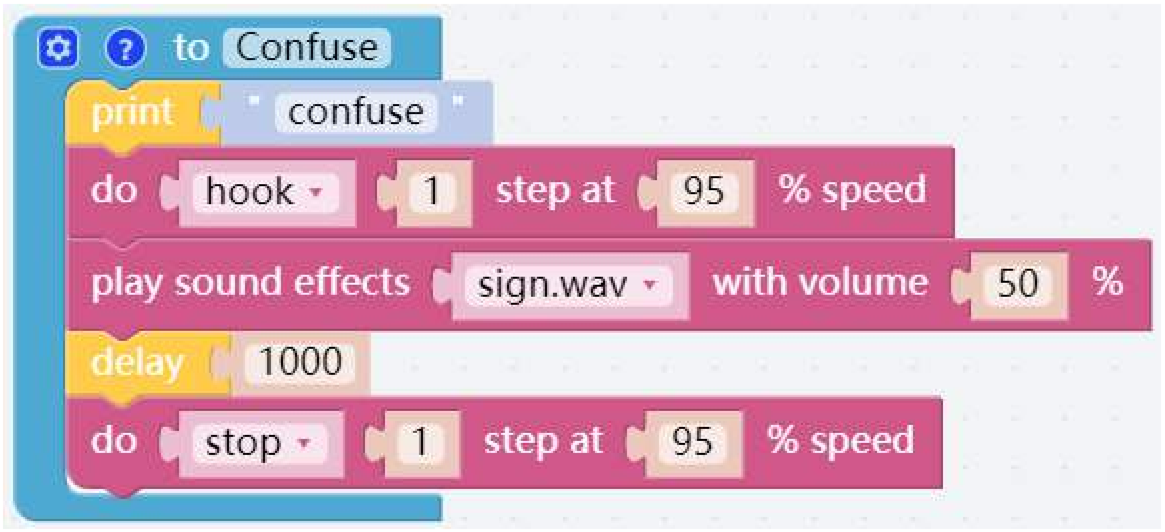
- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.

Shy



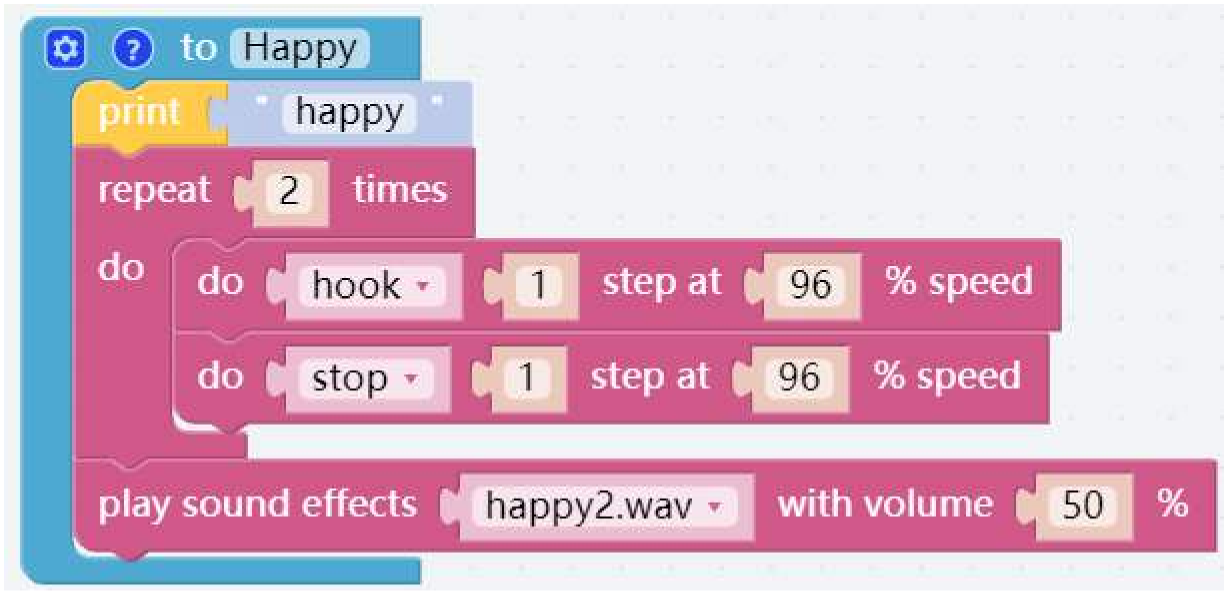
```
to Shy
    print "shy"
    do close 1 step at 95 % speed
    play sound effects talk3.wav with volume 50 %
    delay 1000
    do stop 1 step at 95 % speed
    play sound effects talk2.wav with volume 50 %
    delay 1000
```

Confuse



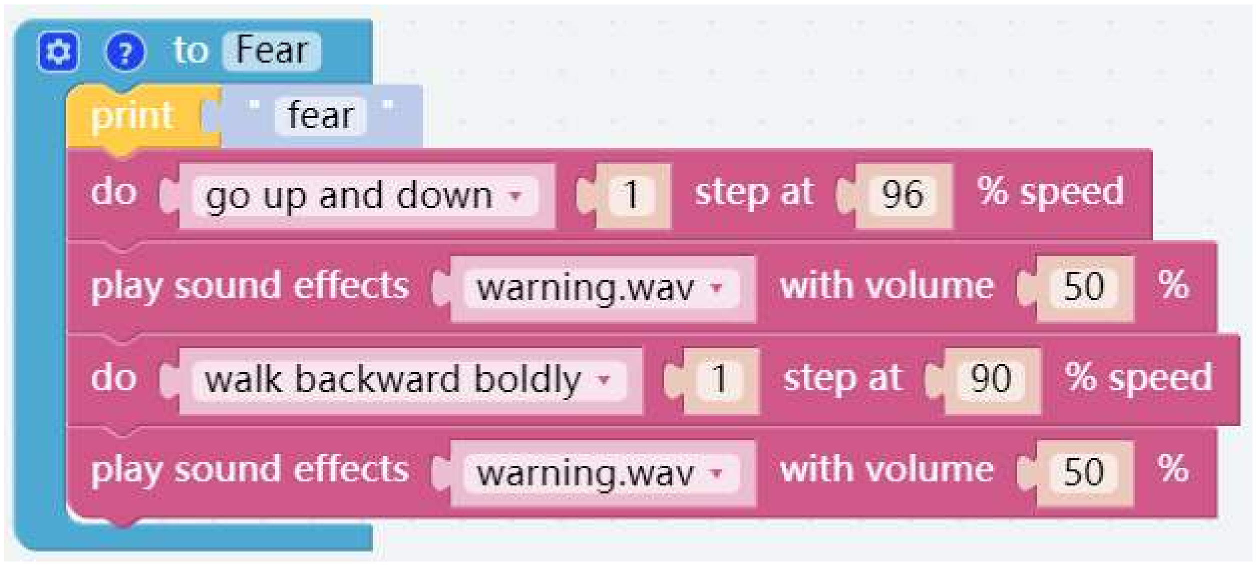
```
to Confuse
    print "confuse"
    do hook 1 step at 95 % speed
    play sound effects sign.wav with volume 50 %
    delay 1000
    do stop 1 step at 95 % speed
```

Happy



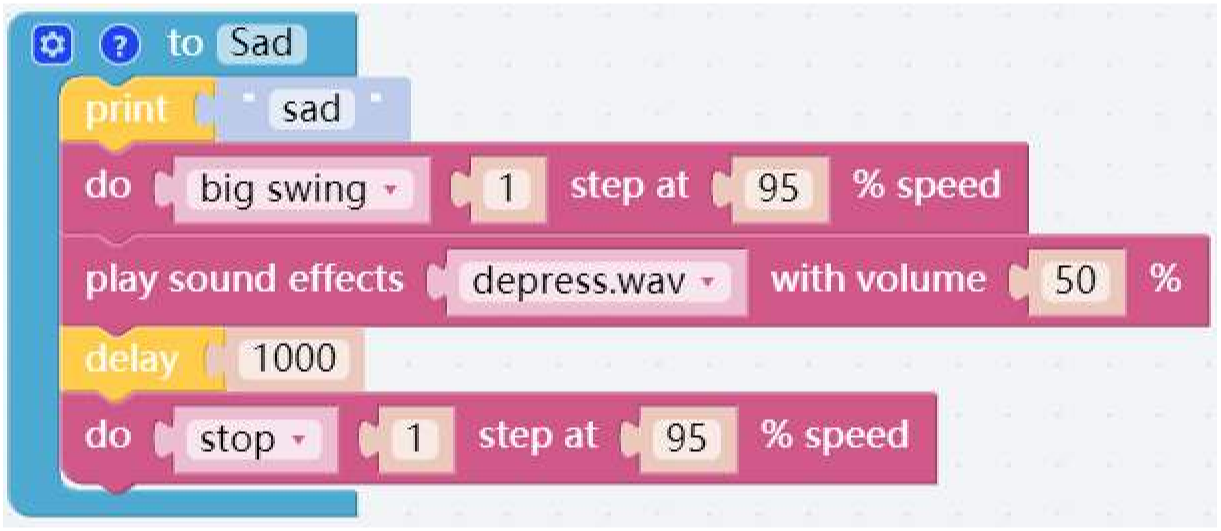
```
to Happy
  print " happy "
  repeat 2 times
  do
    do hook 1 step at 96 % speed
    do stop 1 step at 96 % speed
  play sound effects happy2.wav with volume 50 %
```

Fear



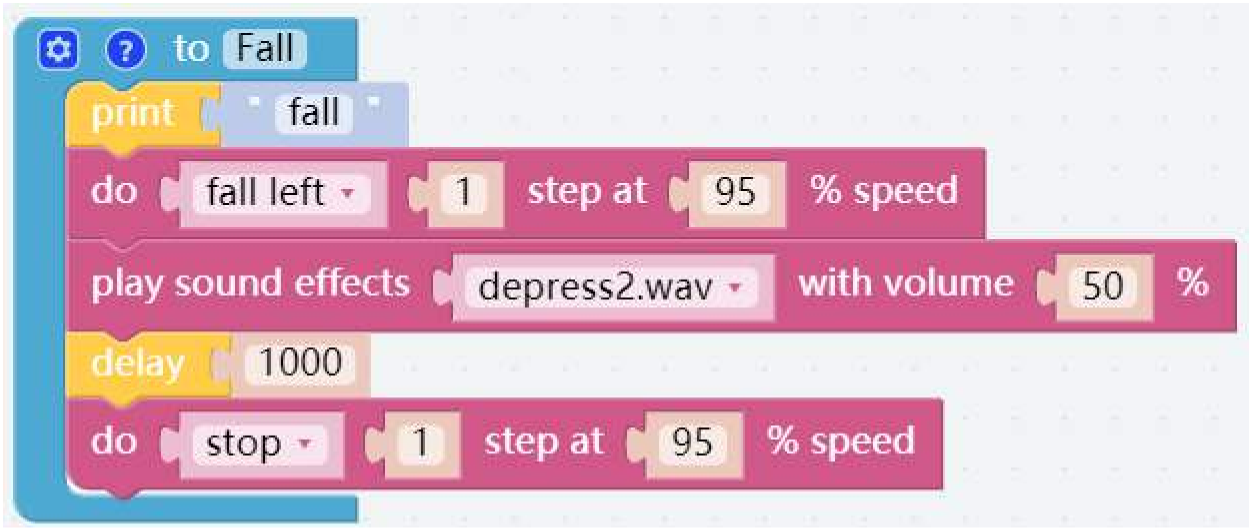
```
to Fear
  print " fear "
  do go up and down 1 step at 96 % speed
  play sound effects warning.wav with volume 50 %
  do walk backward boldly 1 step at 90 % speed
  play sound effects warning.wav with volume 50 %
```

Sad



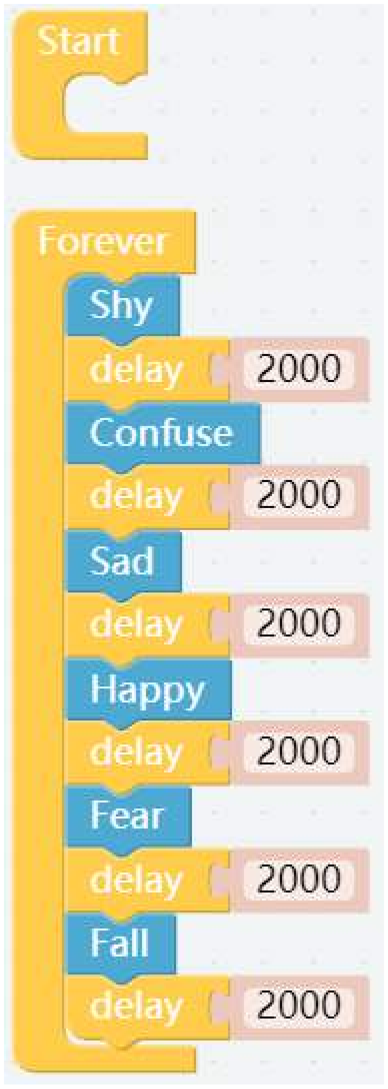
```
to Sad
  print " sad "
  do big swing 1 step at 95 % speed
  play sound effects depress.wav with volume 50 %
  delay 1000
  do stop 1 step at 95 % speed
```

Fall



```
to Fall
  print " fall "
  do fall left 1 step at 95 % speed
  play sound effects depress2.wav with volume 50 %
  delay 1000
  do stop 1 step at 95 % speed
```

Call all custom functions in the Forever block.



```
Start

Forever
    Shy
    delay 2000
    Confuse
    delay 2000
    Sad
    delay 2000
    Happy
    delay 2000
    Fear
    delay 2000
    Fall
    delay 2000
```

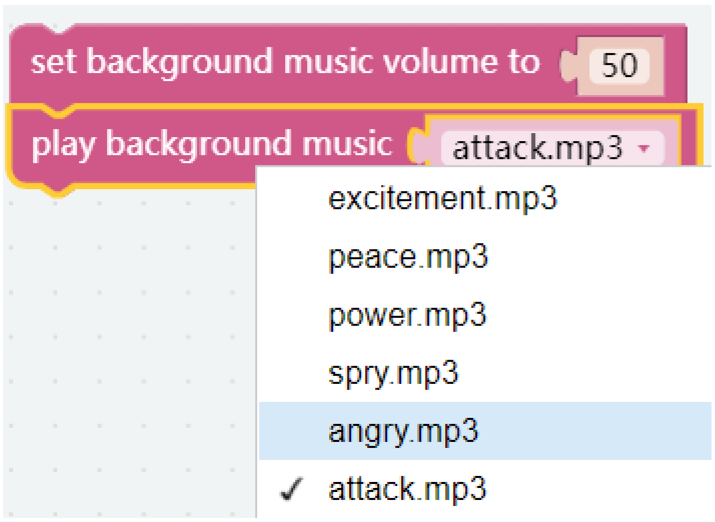
3.6 Dance

Now, PiSltoh will show you his newly learned dance.

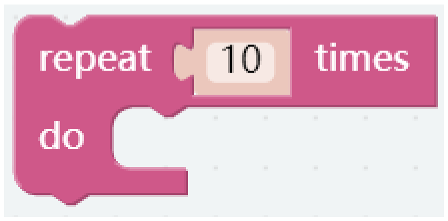
Note: You can download and print the PDF Cartoon Mask for your PiSloth.

TIPS

In addition to having PiSloth play sound effects and speak, it can also play set background music, and the volume of the background music can be adjusted (0%-100%).


set background music volume to 50
play background music attack.mp3
excitement.mp3
peace.mp3
power.mp3
spry.mp3
angry.mp3
✓ attack.mp3

Repeat block can help you execute the same code multiple times to reduce code size.

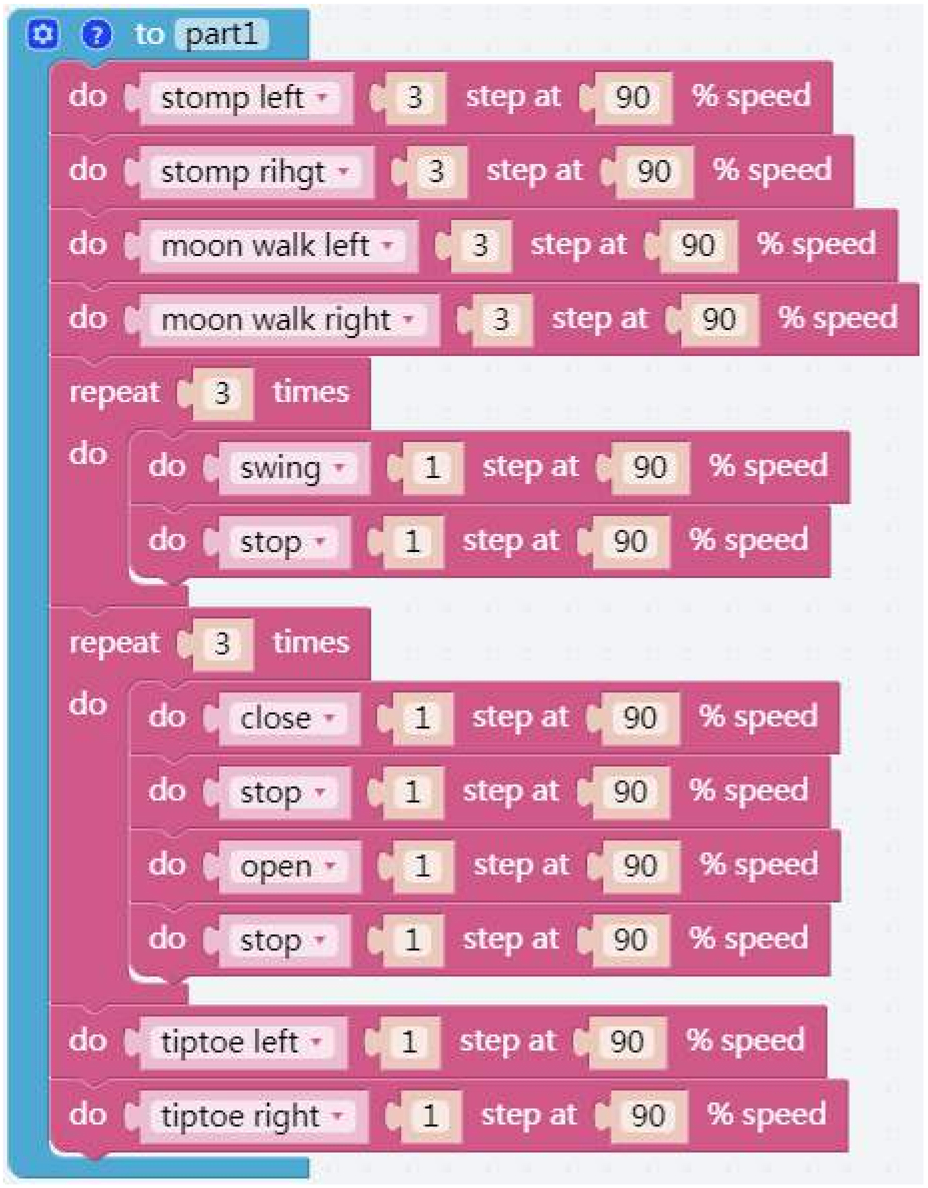


EXAMPLE

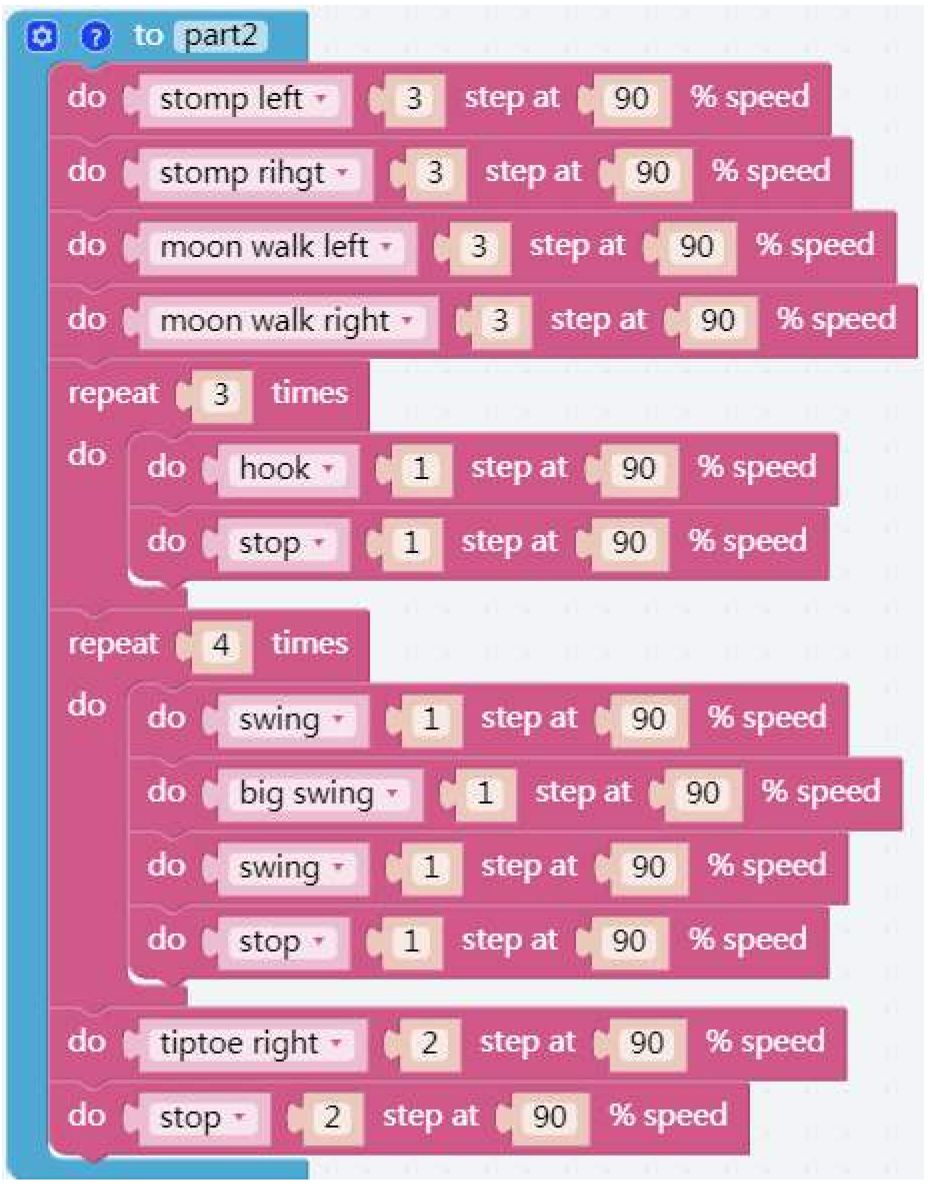
Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.

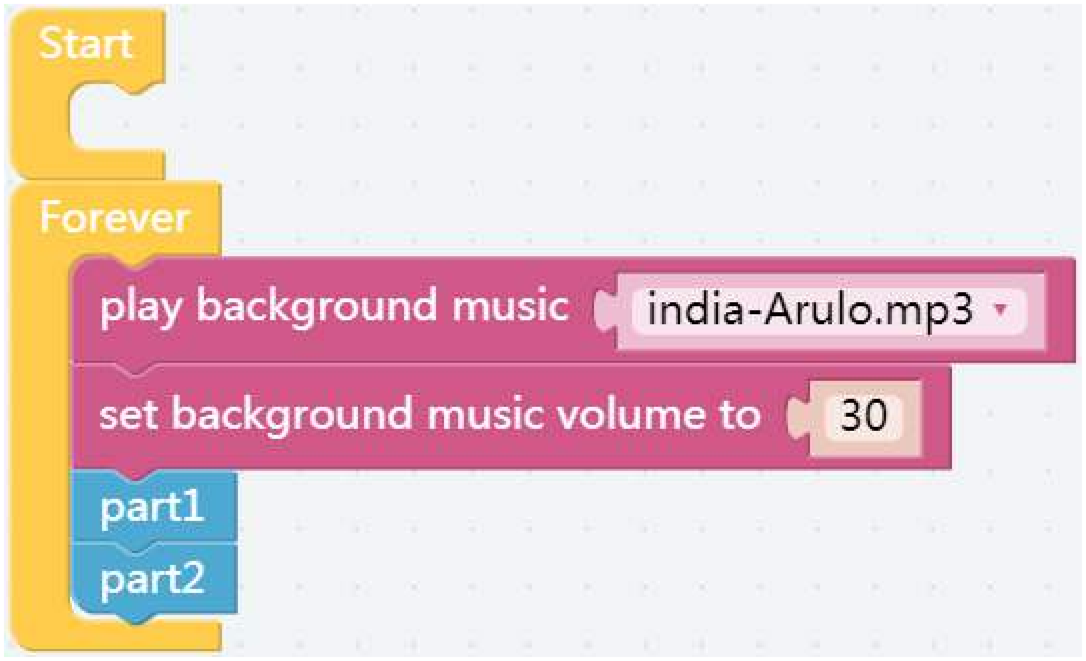
The whole dance is divided into 2 parts, and PiSloth will finish these 2 parts with the music. If you don't pause the code, it will repeat the dance.



```
to part1
do stomp left 3 step at 90 % speed
do stomp rihgt 3 step at 90 % speed
do moon walk left 3 step at 90 % speed
do moon walk right 3 step at 90 % speed
repeat 3 times
do  do swing 1 step at 90 % speed
    do stop 1 step at 90 % speed
repeat 3 times
do  do close 1 step at 90 % speed
    do stop 1 step at 90 % speed
    do open 1 step at 90 % speed
    do stop 1 step at 90 % speed
do tiptoe left 1 step at 90 % speed
do tiptoe right 1 step at 90 % speed
```



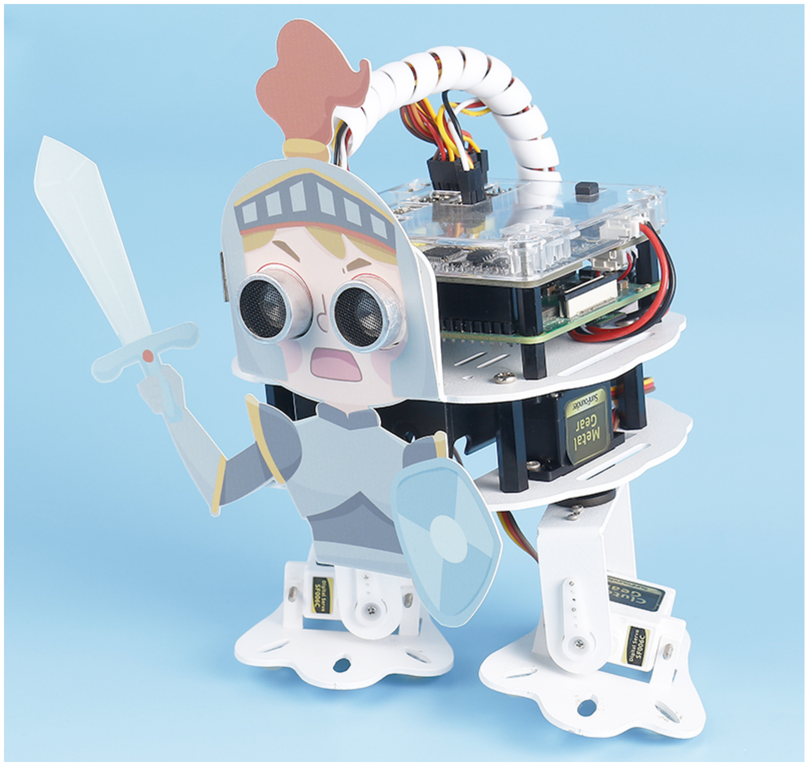
```
to part2
  do stomp left 3 step at 90 % speed
  do stomp rihgt 3 step at 90 % speed
  do moon walk left 3 step at 90 % speed
  do moon walk right 3 step at 90 % speed
  repeat 3 times
  do
    do hook 1 step at 90 % speed
    do stop 1 step at 90 % speed
  repeat 4 times
  do
    do swing 1 step at 90 % speed
    do big swing 1 step at 90 % speed
    do swing 1 step at 90 % speed
    do stop 1 step at 90 % speed
  do tiptoe right 2 step at 90 % speed
  do stop 2 step at 90 % speed
```



```
Start
Forever
    play background music [india-Arulo.mp3 ▾]
    set background music volume to [30]
    part1
    part2
```

3.7 Let's Fight! Warrior!

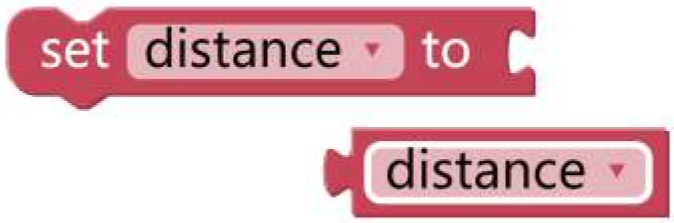
Here, PiSloth is a brave warrior, when it appears in front of the enemy, it will let out a roar and rush to the enemy.



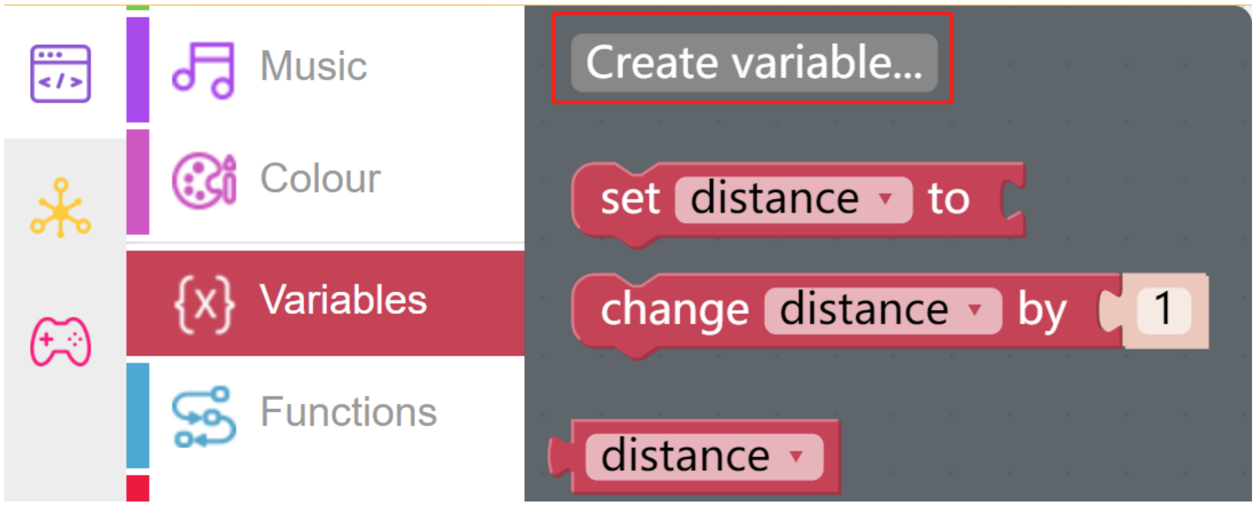
Note: You can download and print the PDF Cartoon Mask for your PiSloth.

TIPS

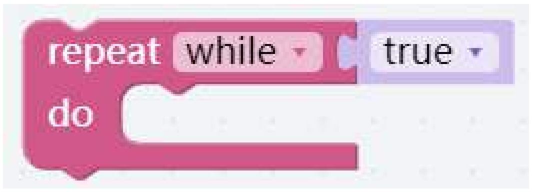
You may want to simplify your program with Variable. For example, when you have multiple functions that need to read the obstacle distance, you don't need to read the value for each function, just load the value into a variable and use it multiple times.



Click the **Create variable** button on the **Variables** category to create a variable named distance.

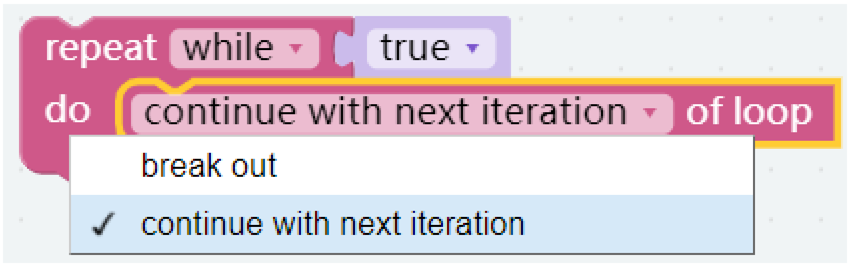


You can use this block to set up an endless loop.



This is a block that jumps out of the loop, and it has two options and can be only used within a loop.

- **break out**: Jump out of the entire loop.
- **continue with next interation**: Jump out of the current loop and enter the next loop.

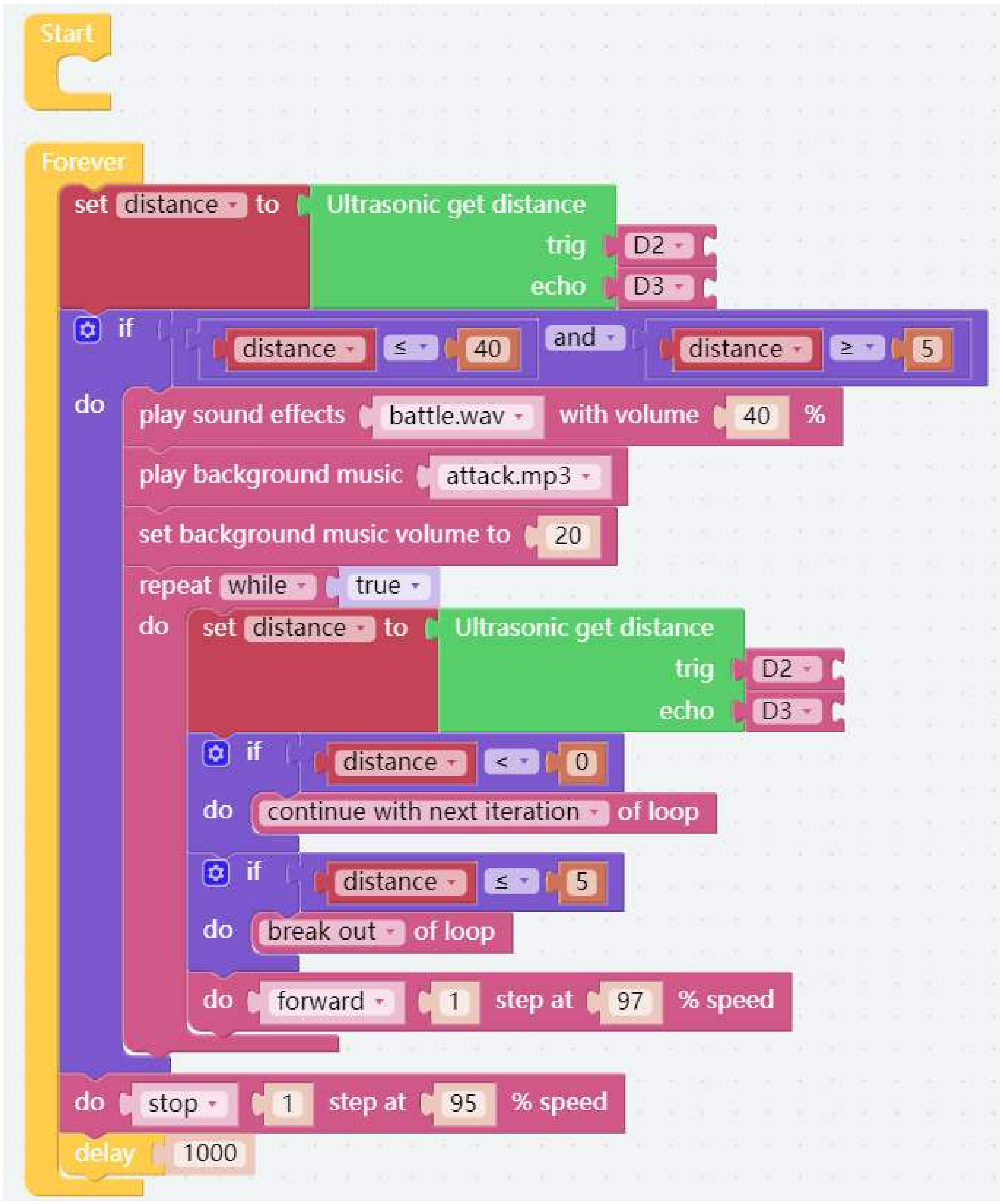

repeat while true
do continue with next iteration of loop
break out
✓ continue with next iteration

EXAMPLE

Note:

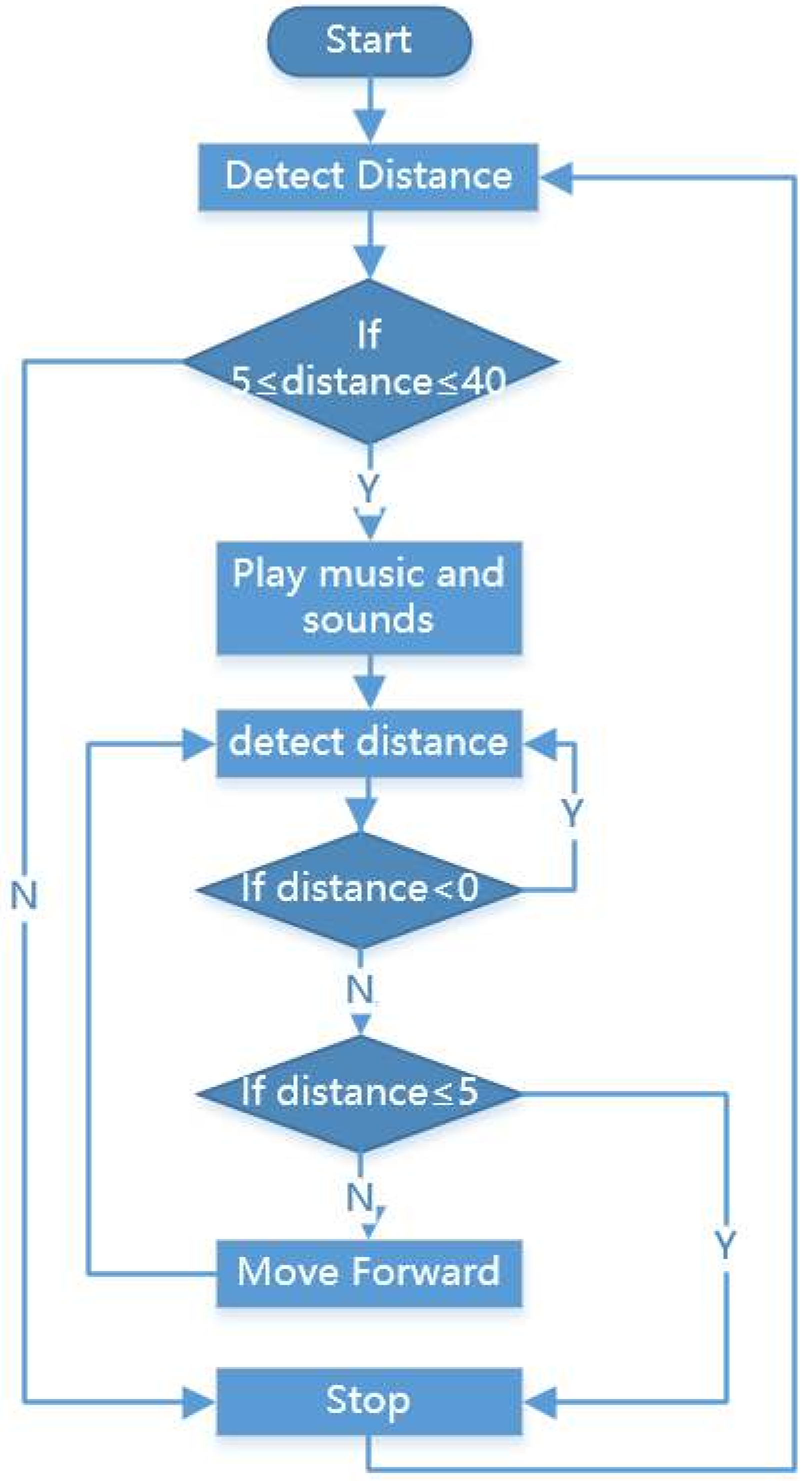
- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.

After the code is run, PiSloth will continuously detect the distance of the obstacle, when the distance is between 5 and 40, PiSloth will make a roaring sound and rush forward; when the distance of the obstacle is less than 5, PiSloth will stop.



```
Start

Forever
  set distance to Ultrasonic get distance
                                   trig D2
                                   echo D3
  if distance ≤ 40 and distance ≥ 5
  do  play sound effects battle.wav with volume 40 %
      play background music attack.mp3
      set background music volume to 20
      repeat while true
      do  set distance to Ultrasonic get distance
                                           trig D2
                                           echo D3
          if distance < 0
          do  continue with next iteration of loop
          if distance ≤ 5
          do  break out of loop
          do forward 1 step at 97 % speed
  do stop 1 step at 95 % speed
  delay 1000
```

Flow Chart


Start
Detect Distance
If
5≤distance≤40
Y
N
Play music and sounds
detect distance
If distance<0
Y
N
If distance≤5
N
Y
Move Forward
Stop

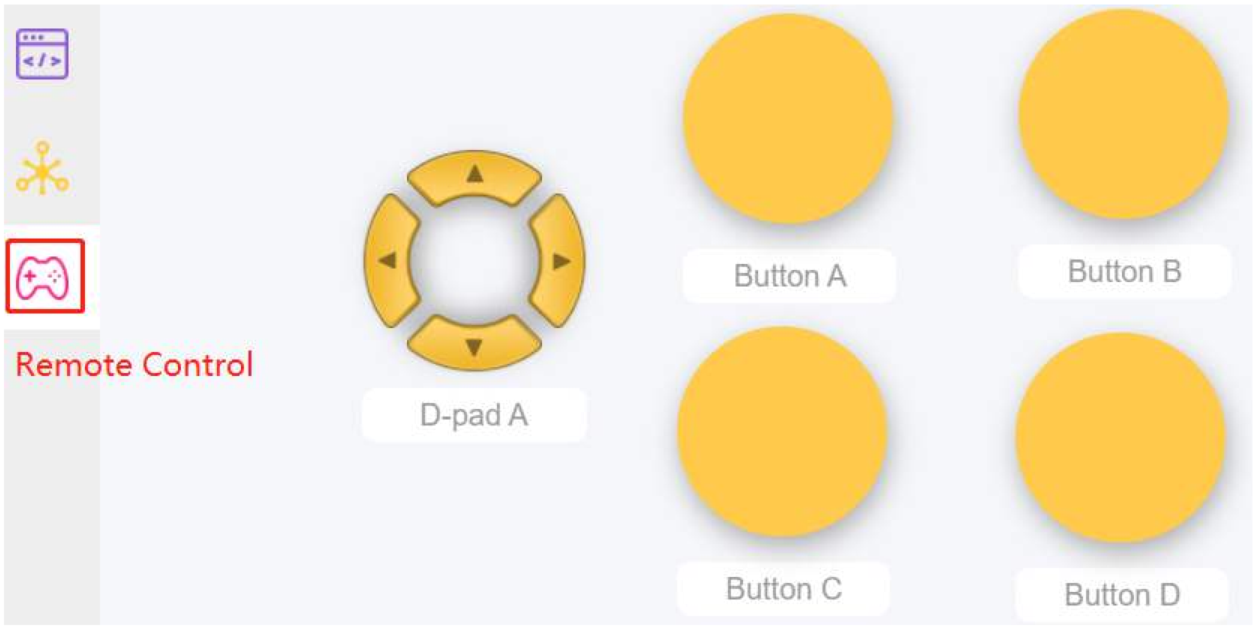
3.8 Remote Control

You can also use the widgets on EzBlock Studio to make PiSloth move.

- How to Use the Remote Control Function?

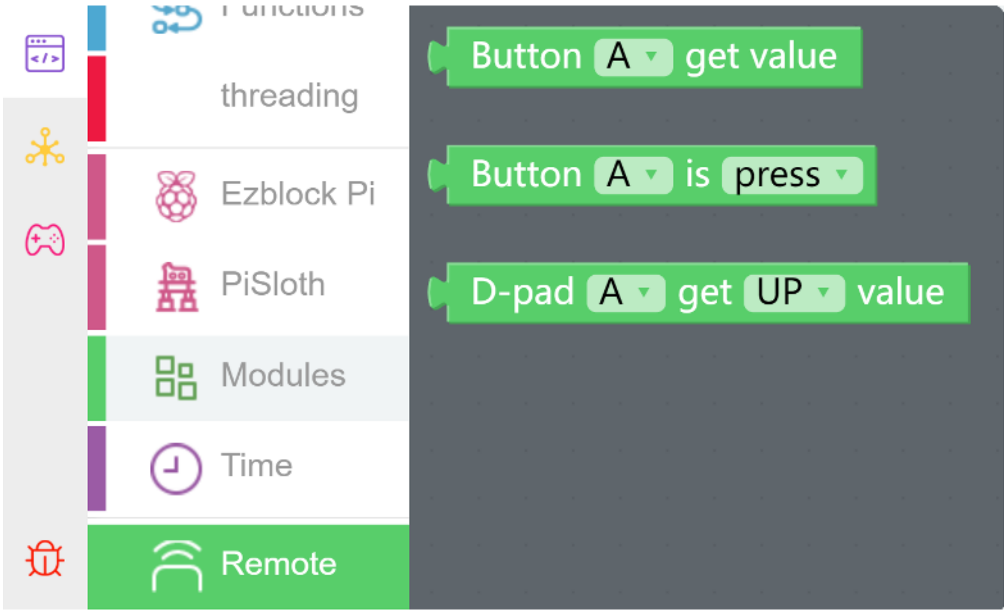
TIPS

To use the remote control function, you need to enter the **Remote Control** page from the left side of main page, and then drag one D-pad and 4 buttons to the central area.



Back in the programming page, you will see an additional Remote category, and the D-pad and Button block appear in it.

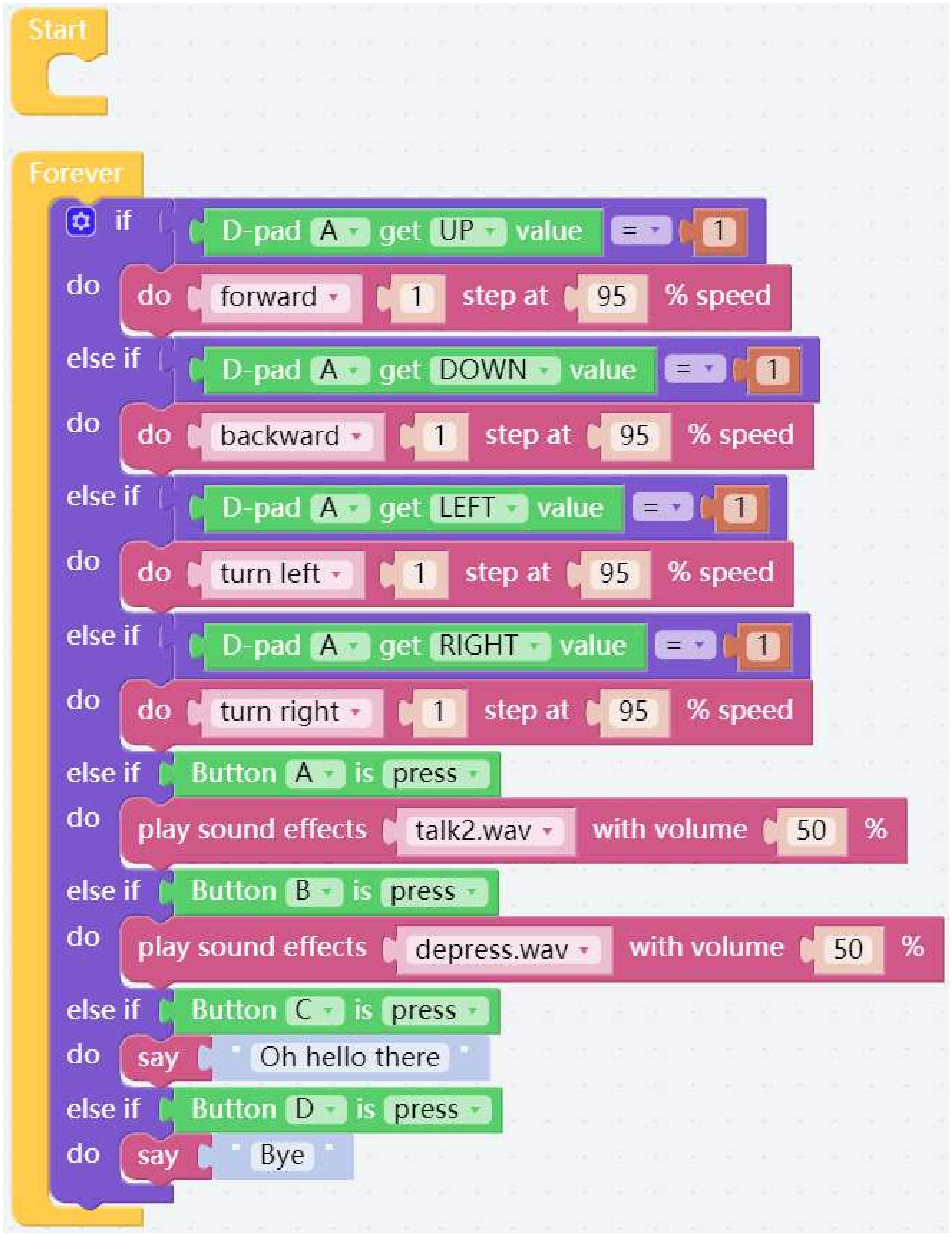
- **Button () get value**: This block is used to read the value of the buttons, if the button is pressed, the value is 1, otherwise it is 0.
- **Button () is (press/release)**: This block and `Button () get value = (0/1)` have the same effect and can be used directly to determine whether a button is pressed or not.
- **D-pad () get () value**: This block is used to read the up/down/left/right (selected through the drop-down menu) pad values, press for 1 and release for 0.



EXAMPLE

Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.



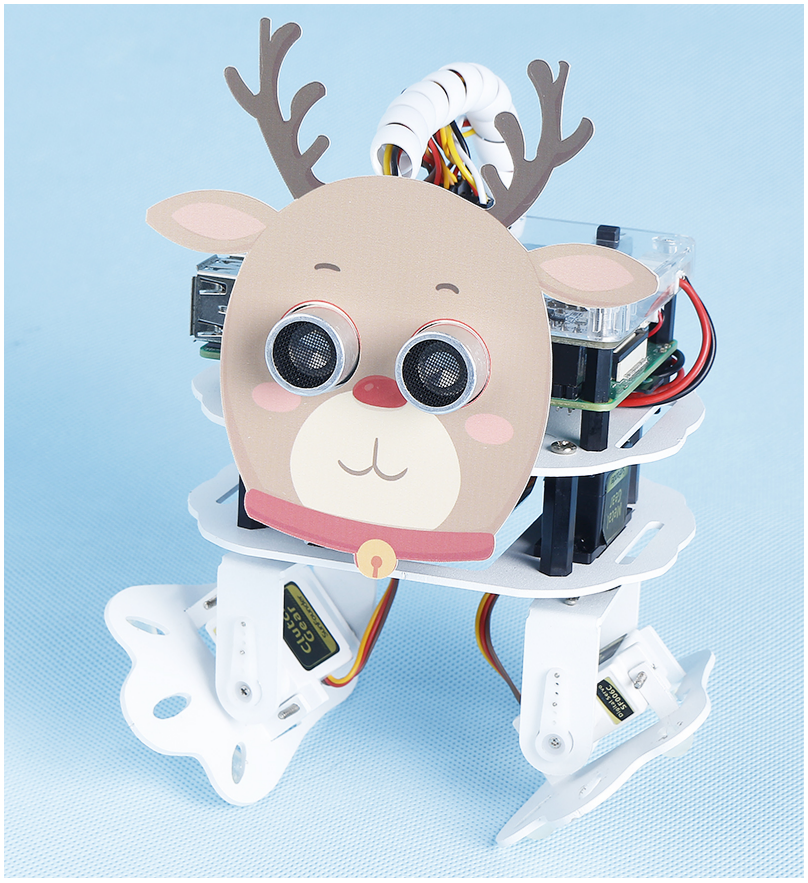
```
Start

Forever
  if   D-pad A get UP value = 1
  do   do forward 1 step at 95 % speed
  else if   D-pad A get DOWN value = 1
  do   do backward 1 step at 95 % speed
  else if   D-pad A get LEFT value = 1
  do   do turn left 1 step at 95 % speed
  else if   D-pad A get RIGHT value = 1
  do   do turn right 1 step at 95 % speed
  else if   Button A is press
  do   play sound effects talk2.wav with volume 50 %
  else if   Button B is press
  do   play sound effects depress.wav with volume 50 %
  else if   Button C is press
  do   say " Oh hello there "
  else if   Button D is press
  do   say " Bye "
```

3.9 Custom Step

In the previous projects, we used a lot of actions that we wrote, so how are these actions composed and done? Generally speaking, an action is composed of one or more steps.

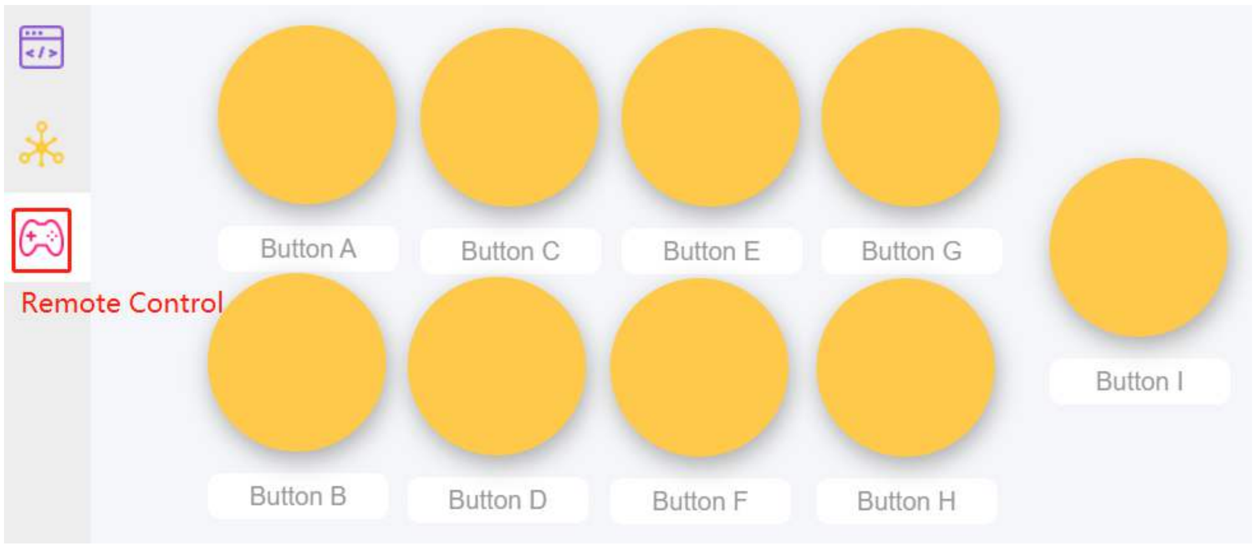
In this project, we will learn how to customize PiSloth's step.

All we have to do is to use the buttons in the remote control page to make PiSloth complete the step shown in the figure below, and then get the angles of the 4 Servos at that time.

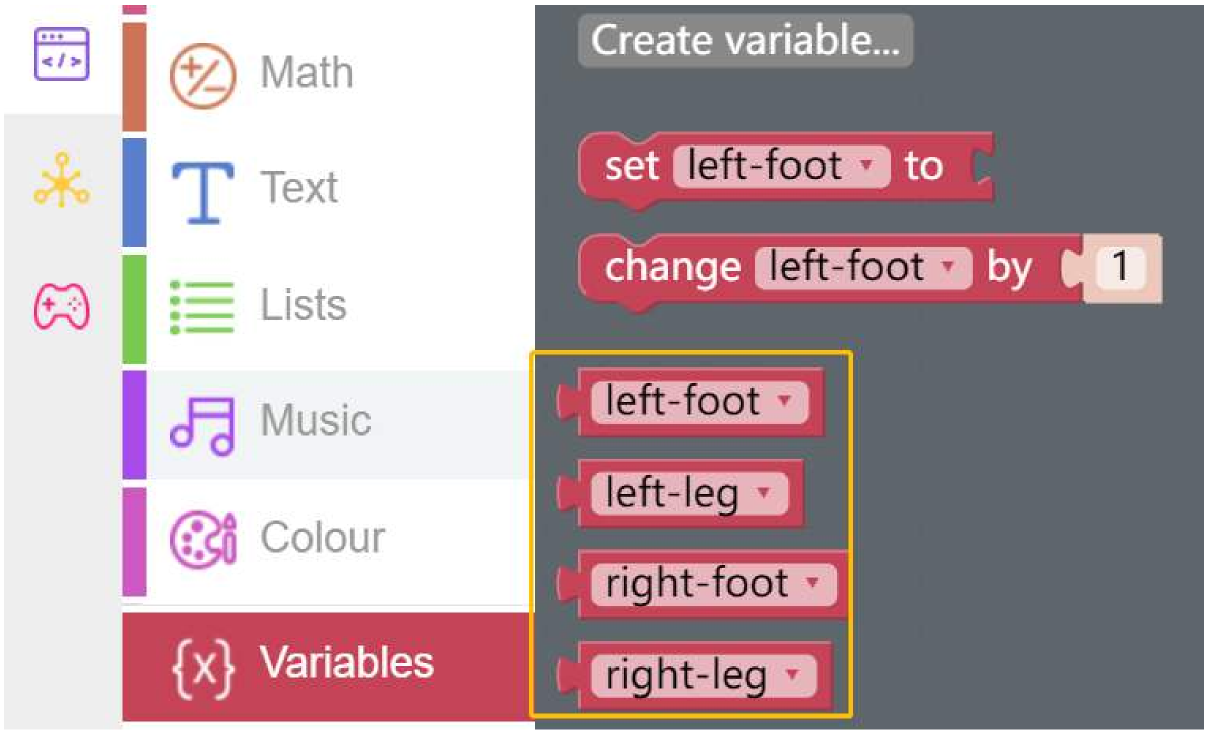


Note: You can download and print the PDF Cartoon Mask for your PiSloth.

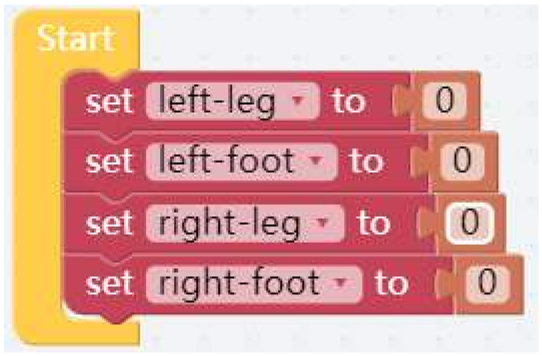
Step 1: Drag out 9 buttons in the Remote Control to control the rotation angles of the 4 Servos on the PiSloth.



Step 2: Create 4 variables to store the angles of the 4 Servos.



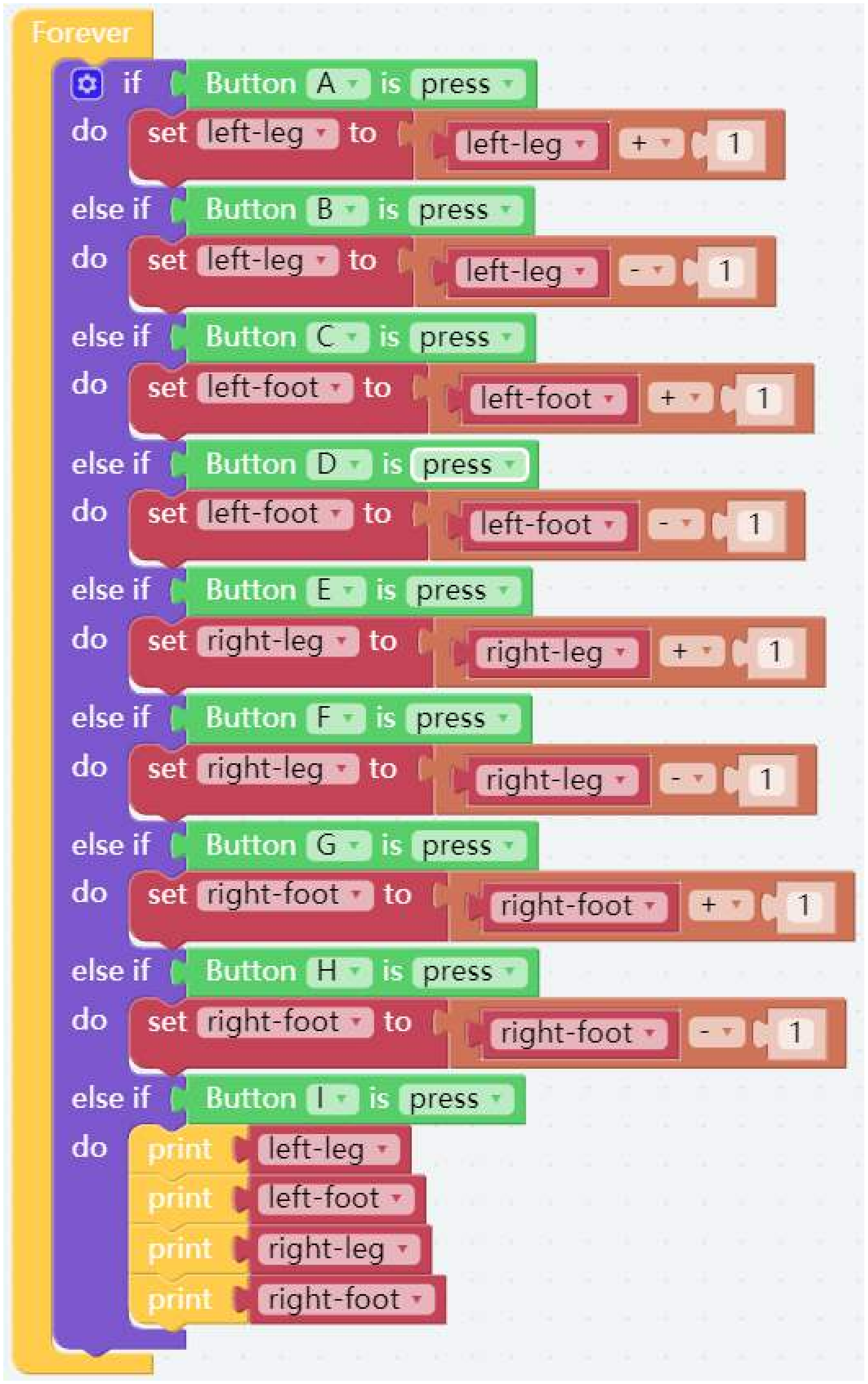
Then initialize the angle to 0.



```
set left-leg to 0
set left-foot to 0
set right-leg to 0
set right-foot to 0
```

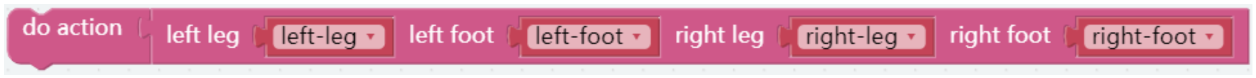
Step 3: Reads the values of the different buttons that are used to control the angles of the Servos.

- **button AB** control the **left-leg**.
- **button CD** control the **left-foot**.
- **button EF** control the **right-leg**.
- **button GH** control the **right-foot**.
- Press **button I** and the angles of the 4 Servos will be printed in the Debug Monitor.

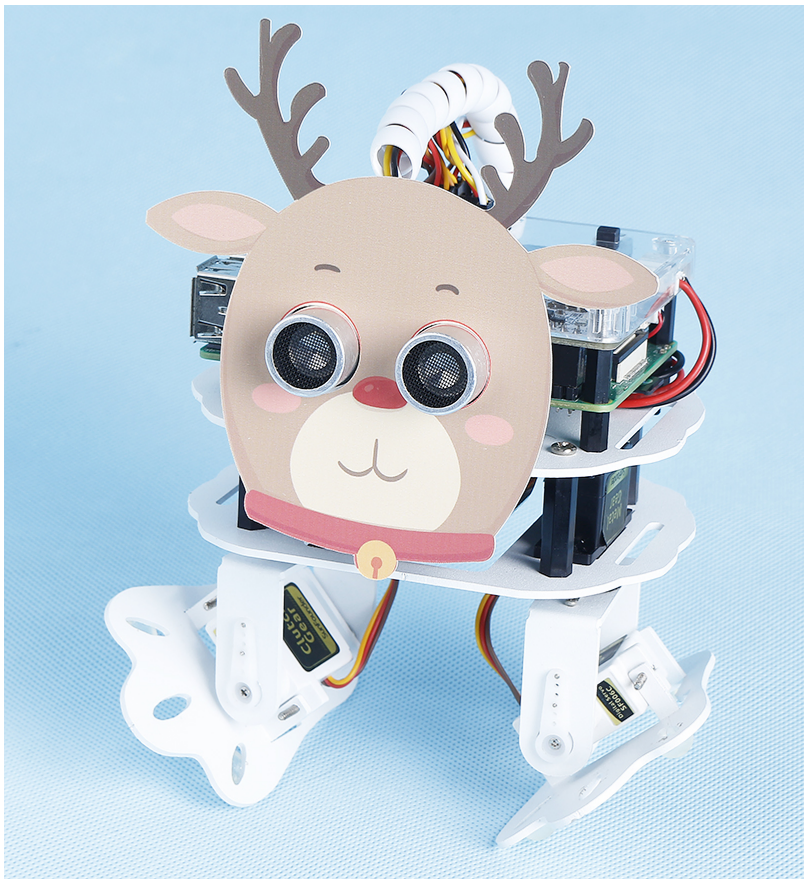


```
Forever
  if Button A is press
  do set left-leg to left-leg + 1
  else if Button B is press
  do set left-leg to left-leg - 1
  else if Button C is press
  do set left-foot to left-foot + 1
  else if Button D is press
  do set left-foot to left-foot - 1
  else if Button E is press
  do set right-leg to right-leg + 1
  else if Button F is press
  do set right-leg to right-leg - 1
  else if Button G is press
  do set right-foot to right-foot + 1
  else if Button H is press
  do set right-foot to right-foot - 1
  else if Button I is press
  do print left-leg
     print left-foot
     print right-leg
     print right-foot
```

Step 4: At the end of the Forever block, fill in the angle values read into the 4 servos and use the **do action** block to make PiSloth do this step.

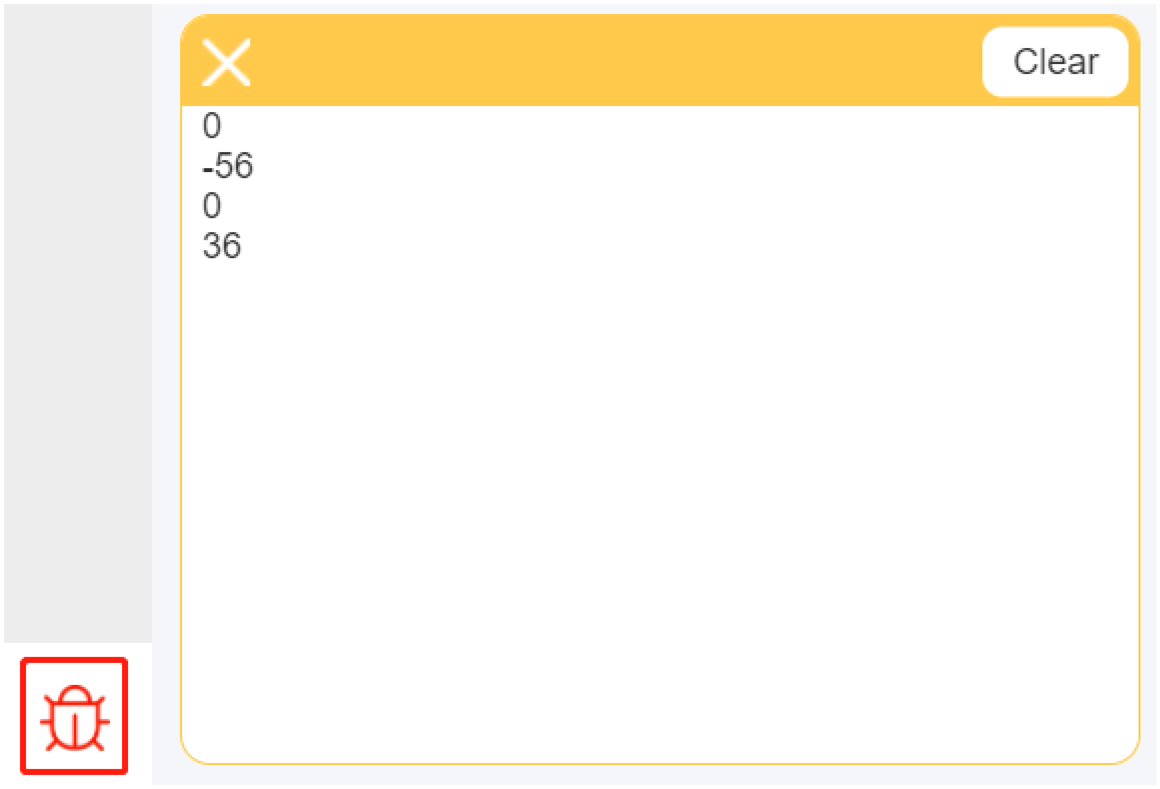


Step 5: Once the code is complete, click the **download** icon in the bottom right corner to download and run the code. Now we can click **button CD** and **button GH** (according to the actual code) to make PiSloth pose like this, you can also make it do other steps.



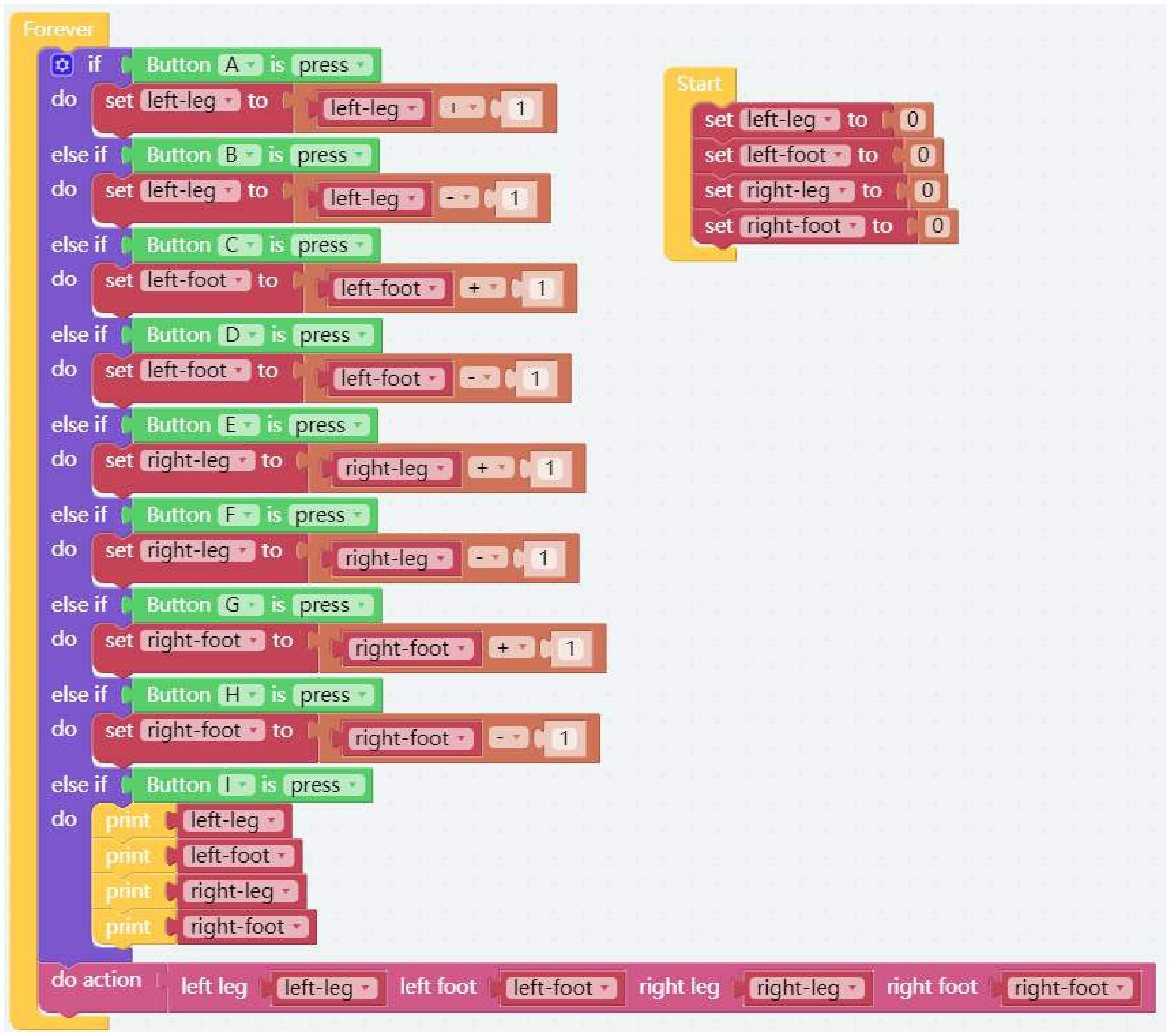
Step 6: Click on the Debug Monitor icon in the bottom left corner, and you will see the angle of the 4 servos in the Debug Monitor at that moment when you press **button I**.

Note: Some times more than 2 sets of data may appear because if you click **button I** for a little longer, EzBlock will think **button I** was clicked 2 times. You can clear the data and click button I again.



```
0
-56
0
36
```

The complete code is as follows:



```
Forever
  if Button A is press
  do set left-leg to left-leg + 1
  else if Button B is press
  do set left-leg to left-leg - 1
  else if Button C is press
  do set left-foot to left-foot + 1
  else if Button D is press
  do set left-foot to left-foot - 1
  else if Button E is press
  do set right-leg to right-leg + 1
  else if Button F is press
  do set right-leg to right-leg - 1
  else if Button G is press
  do set right-foot to right-foot + 1
  else if Button H is press
  do set right-foot to right-foot - 1
  else if Button I is press
  do print left-leg
     print left-foot
     print right-leg
     print right-foot
  do action left leg left-leg left foot left-foot right leg right-leg right foot right-foot

Start
  set left-leg to 0
  set left-foot to 0
  set right-leg to 0
  set right-foot to 0
```

3.10 Custom Action

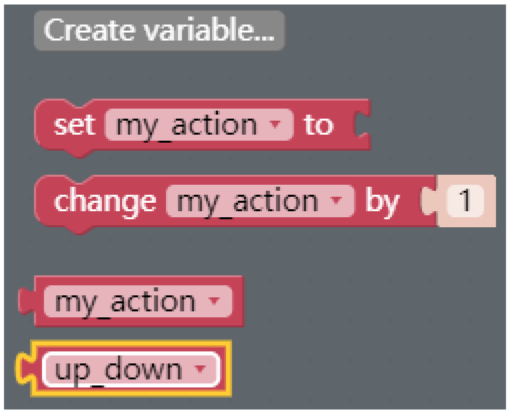
In the previous project, we were able to give PiSloth custom steps, so how do we combine these steps into actions?

For example, have PiSloth make the step from the previous project and then return to the initial position.

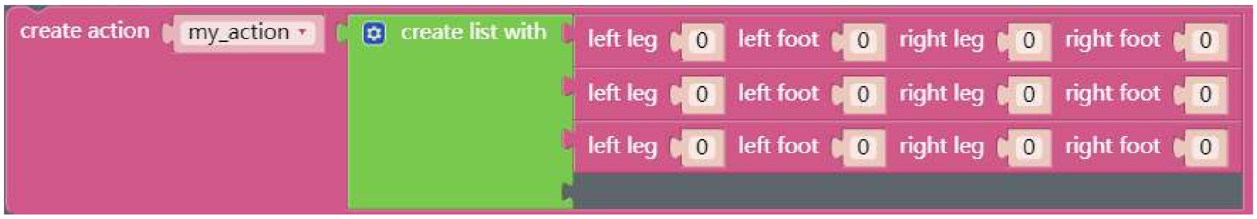
Note: You can download and print the PDF Cartoon Mask for your PiSloth.

TIPS

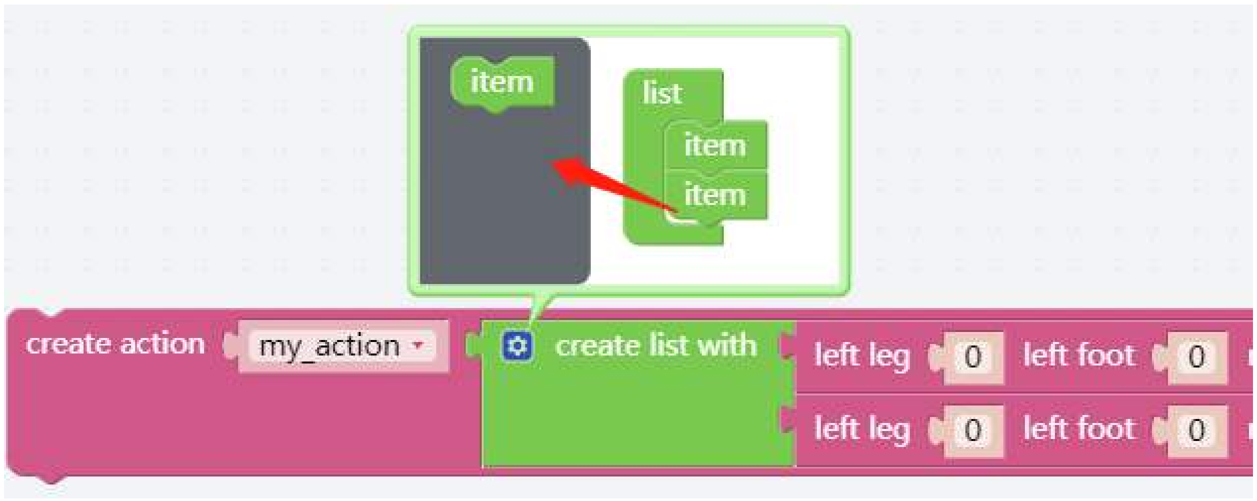
Create a variable **up_down** to store this action.



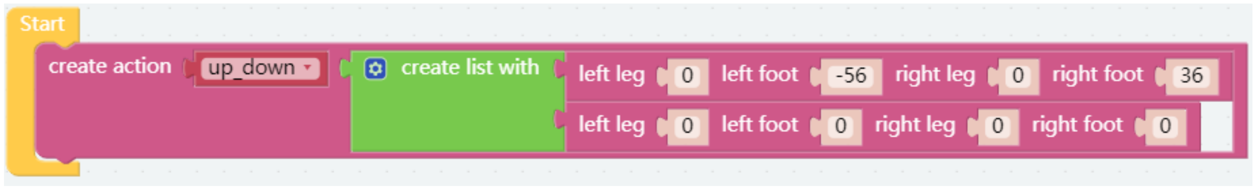
You can use the **create action** block to make PiSloth do this action. These values represent the angles of the 4 Servos on the PiSloth. The range is (-90~90).


create action
my_action
create list with
left leg 0 left foot 0 right leg 0 right foot 0
left leg 0 left foot 0 right leg 0 right foot 0
left leg 0 left foot 0 right leg 0 right foot 0

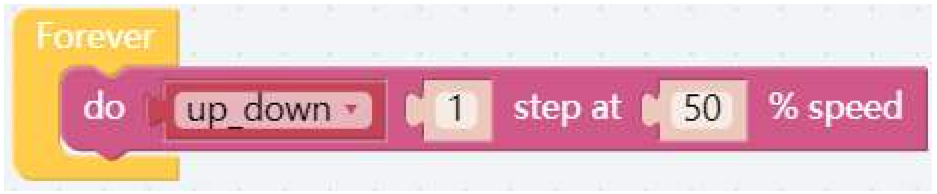
Here you can increase or decrease the number of items by dragging it.



Fill in the angle obtained in the previous project and name this action **up_down** (drag it from **Variables** category).


Start
create action
up_down
create list with
left leg 0 left foot -56 right leg 0 right foot 36
left leg 0 left foot 0 right leg 0 right foot 0

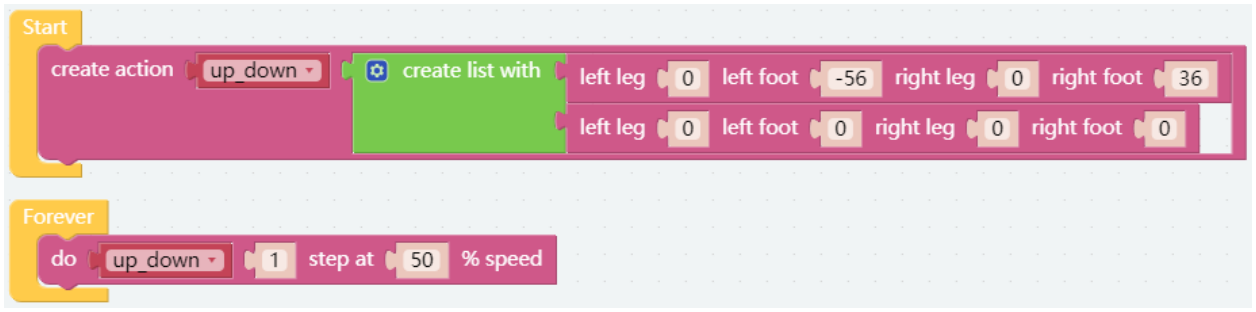
Use the **do** block to make PiSloth do this action once at 50% speed.



EXAMPLE

Note:

- You can write the program according to the following picture, please refer to the tutorial: How to Create a New Project?
- Or find the code with the same name on the Examples page of the EzBlock Studio and click Run or Edit directly.


Start
create action
up_down
create list with
left leg
0
left foot
-56
right leg
0
right foot
36
left leg
0
left foot
0
right leg
0
right foot
0
Forever
do
up_down
1
step at
50
% speed

CHAPTER

FOUR

PLAY WITH PYTHON

If you want to program in python, then you will need to learn some basic Python programming skills and basic knowledge of Raspberry Pi, please configure the Raspberry Pi first according to *Quick Guide on Python*.

4.1 Quick Guide on Python

This section is to teach you how to install Raspberry Pi OS, configure wifi to Raspberry Pi, remote access to Raspberry Pi to run the corresponding code.

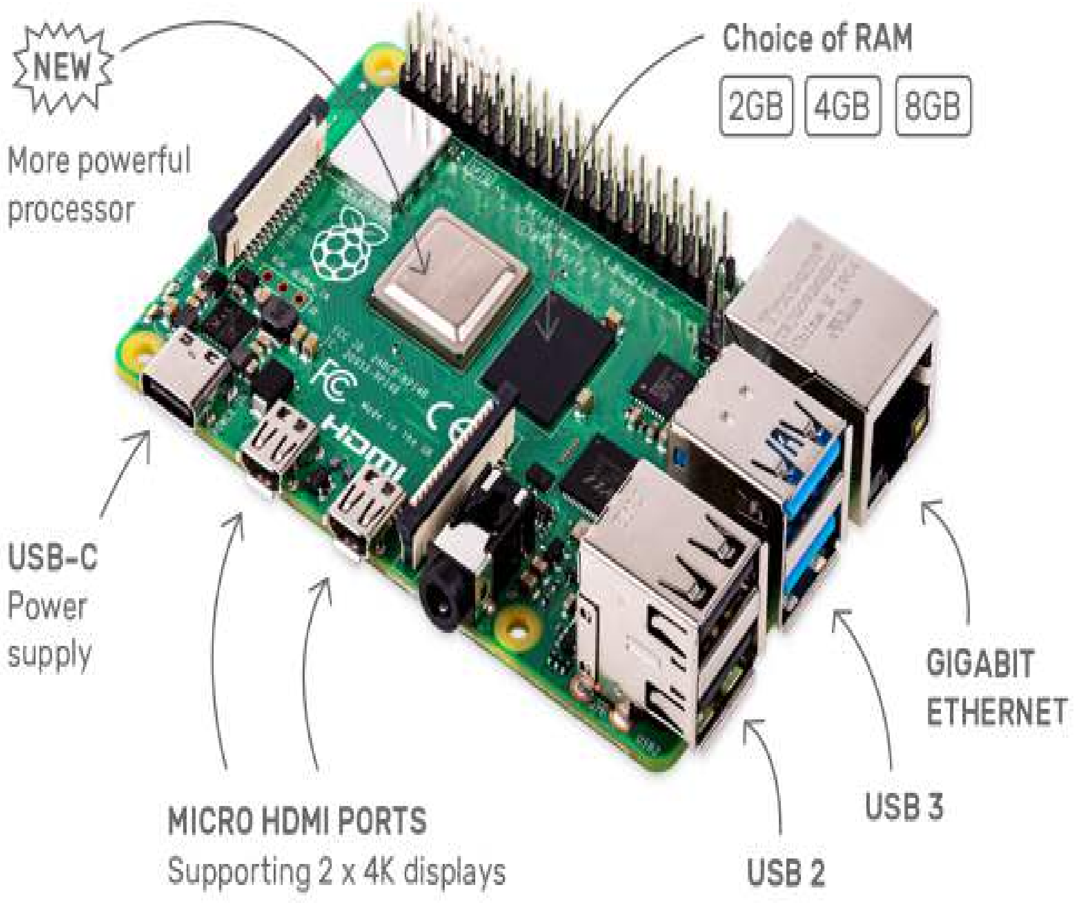
If you are familiar with Raspberry Pi and can open the command line successfully, then you can skip the first 3 parts and then complete the last part.

4.1.1 What Do We Need?

Required Components

Raspberry Pi

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.



Power Adapter

To connect to a power socket, the Raspberry Pi has a micro USB port (the same found on many mobile phones). You will need a power supply which provides at least 2.5 amps.

Micro SD Card

Your Raspberry Pi needs an Micro SD card to store all its files and the Raspberry Pi OS. You will need a micro SD card with a capacity of at least 8 GB

Optional Components

Screen

To view the desktop environment of Raspberry Pi, you need to use the screen that can be a TV screen or a computer monitor. If the screen has built-in speakers, the Pi plays sounds via them.

Mouse & Keyboard

When you use a screen , a USB keyboard and a USB mouse are also needed.

HDMI

The Raspberry Pi has a HDMI output port that is compatible with the HDMI ports of most modern TV and computer monitors. If your screen has only DVI or VGA ports, you will need to use the appropriate conversion line.

Case

You can put the Raspberry Pi in a case; by this means, you can protect your device.

Sound or Earphone

The Raspberry Pi is equipped with an audio port about 3.5 mm that can be used when your screen has no built-in speakers or when there is no screen operation.

4.1.2 Installing the OS

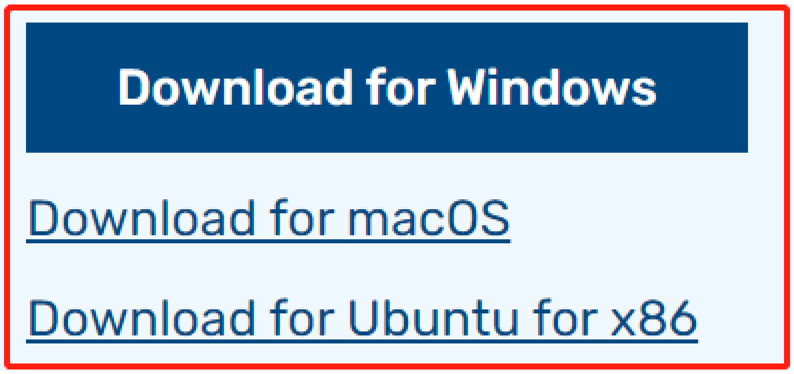
Required Components

Any Raspberry Pi	1 * Personal Computer
1 * Micro SD card	

Step 1

Raspberry Pi have developed a graphical SD card writing tool that works on Mac OS, Ubuntu 18.04 and Windows, and is the easiest option for most users as it will download the image and install it automatically to the SD card.

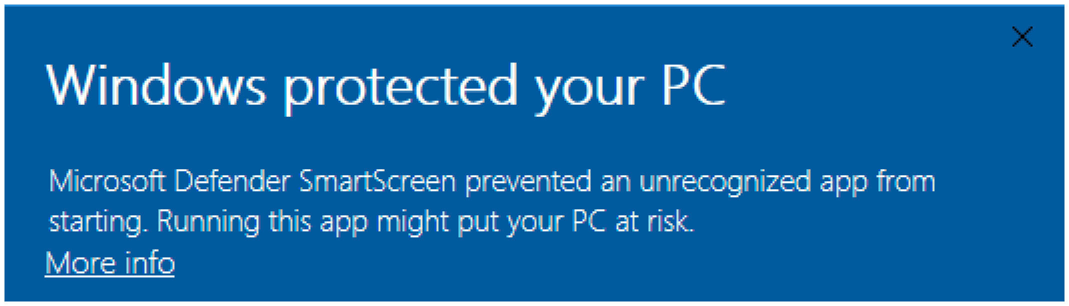
Visit the download page: https://www.raspberrypi.org/software/. Click on the link for the Raspberry Pi Imager that matches your operating system, when the download finishes, click it to launch the installer.



Step 2

When you launch the installer, your operating system may try to block you from running it. For example, on Windows I receive the following message:

If this pops up, click on **More info** and then **Run anyway**, then follow the instructions to install the Raspberry Pi Imager.



Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.
More info

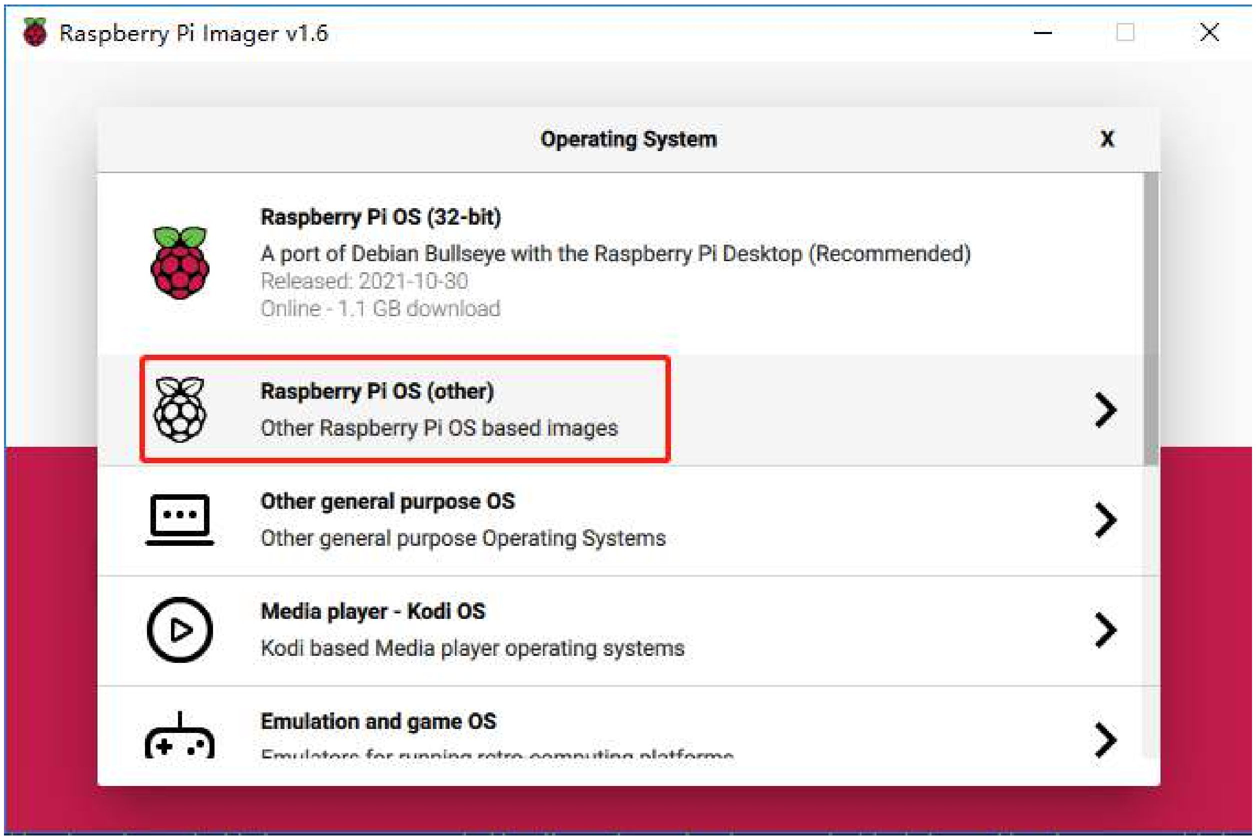
Step 3

Insert your SD card into the computer or laptop SD card slot.

Step 4

Warning: Upgrading the Raspberry Pi OS to **Debian Bullseye** will cause some features to not work, so it is recommended to continue using the **Debian Buster** version.

In the Raspberry Pi Imager, click **CHOOSE OS** -> **Raspberry Pi OS(other)**.



Raspberry Pi Imager v1.6

Operating System

Raspberry Pi OS (32-bit)
A port of Debian Bullseye with the Raspberry Pi Desktop (Recommended)
Released: 2021-10-30
Online - 1.1 GB download

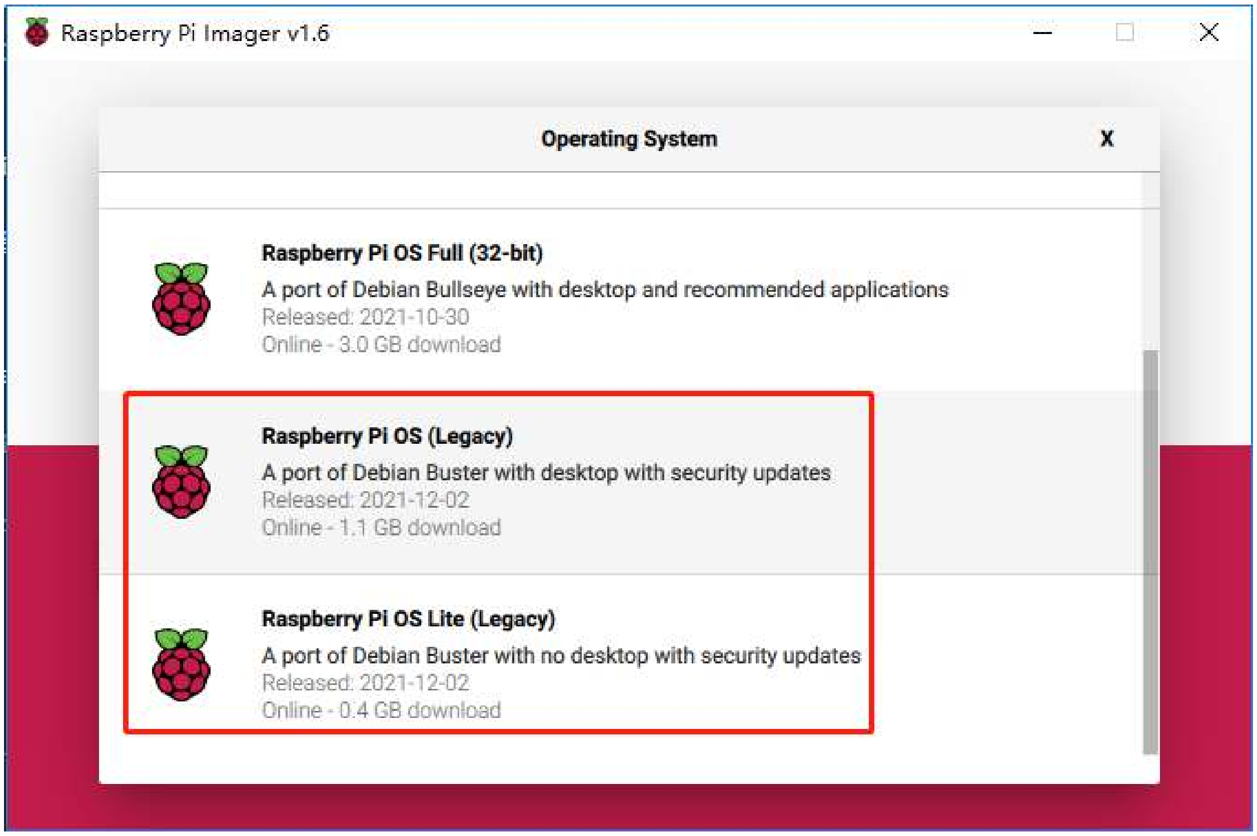
Raspberry Pi OS (other)
Other Raspberry Pi OS based images

Other general purpose OS
Other general purpose Operating Systems

Media player - Kodi OS
Kodi based Media player operating systems

Emulation and game OS

Scroll down to the end of the newly opened page and you will see **Raspberry Pi OS(Legacy)** and **Raspberry Pi OS Lite(Legacy)**, these are security updates for Debian Buster, the difference between them is with or without the desktop. It is recommended to install **Raspberry Pi OS(Legacy)**, the system with the desktop.



Raspberry Pi Imager v1.6

Operating System

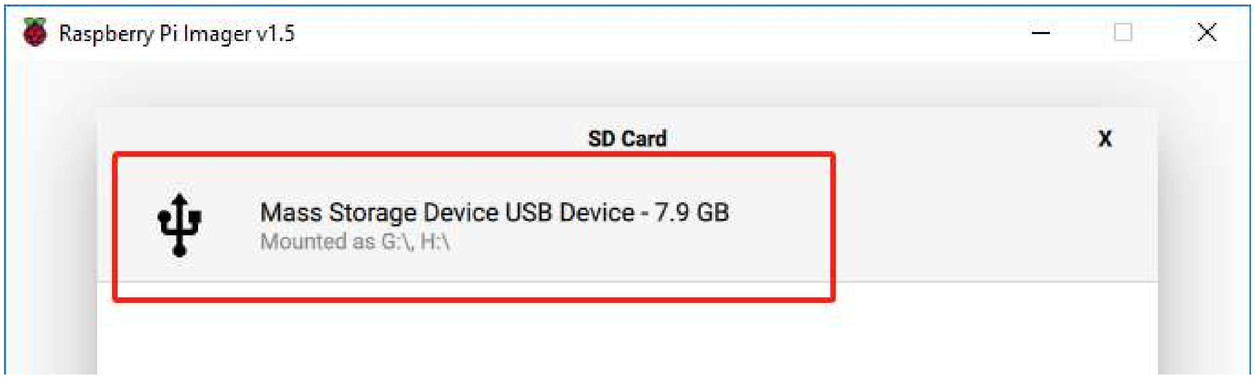
Raspberry Pi OS Full (32-bit)
A port of Debian Bullseye with desktop and recommended applications
Released: 2021-10-30
Online - 3.0 GB download

Raspberry Pi OS (Legacy)
A port of Debian Buster with desktop with security updates
Released: 2021-12-02
Online - 1.1 GB download

Raspberry Pi OS Lite (Legacy)
A port of Debian Buster with no desktop with security updates
Released: 2021-12-02
Online - 0.4 GB download

Step 5

Select the SD card you are using.



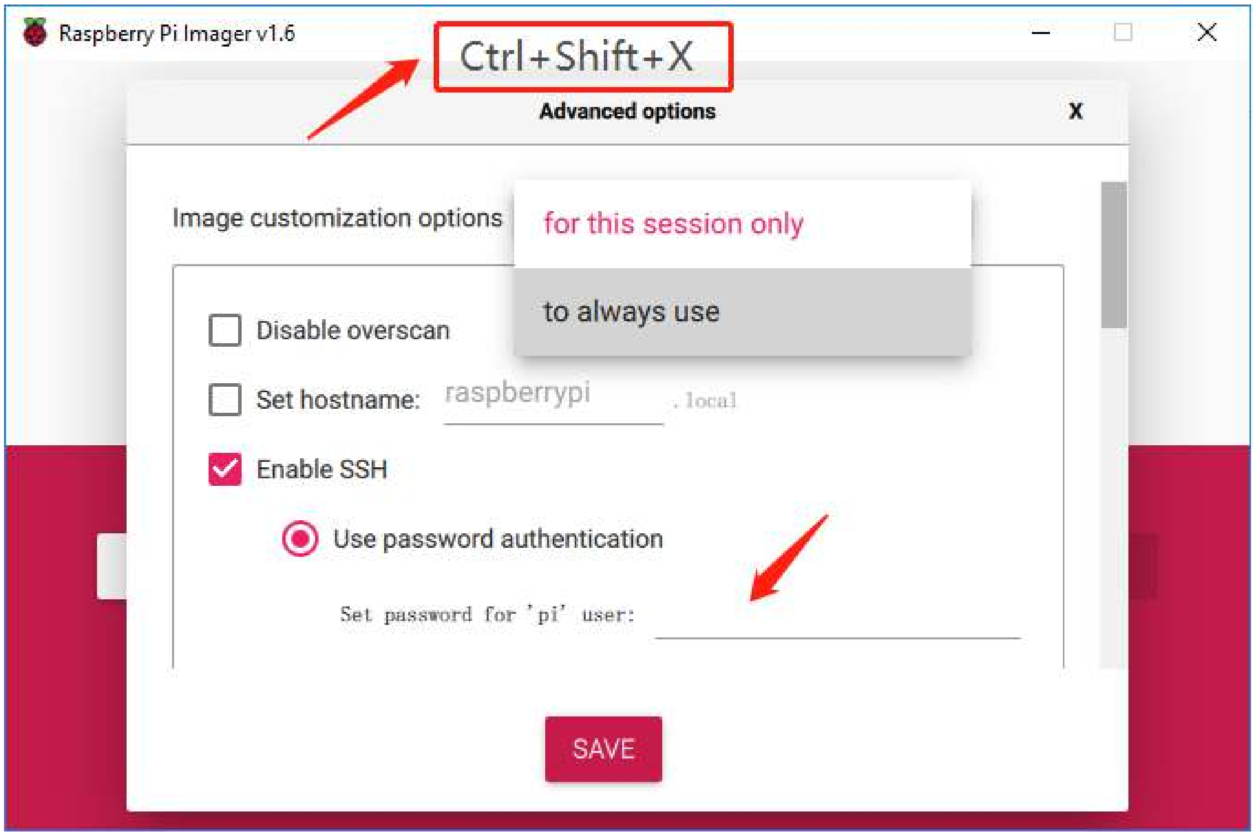
Raspberry Pi Imager v1.5

SD Card

Mass Storage Device USB Device - 7.9 GB
Mounted as G:\, H:\

Step 6

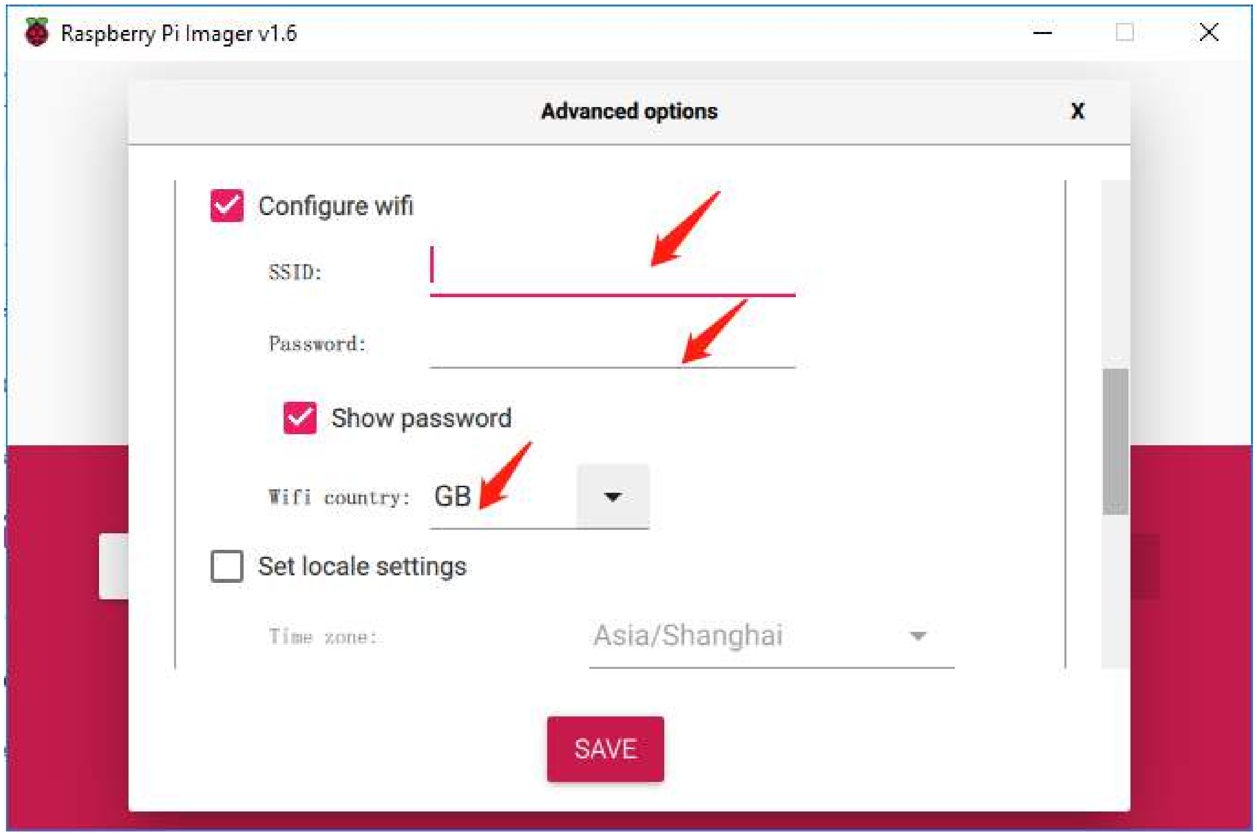
Press **Ctrl+Shift+X** or click the **setting** button to open the **Advanced options** page to enable SSH and configure wifi, these 2 items must be set, the others depend on your choice . You can choose to always use this image customization options.



- [ ] Disable overscan
- [ ] Set hostname: raspberrypi .local
- [x] Enable SSH
  - ● Use password authentication
  - Set password for 'pi' user:

Then scroll down to complete the wifi configuration and click **SAVE**.

Note: **wifi country** should be set the two-letter ISO/IEC alpha2 code for the country in which you are using your Raspberry Pi, please refer to the following link: https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2#Officially_assigned_code_elements



Configure wifi [✓]

SSID:

Password:

Show password [✓]

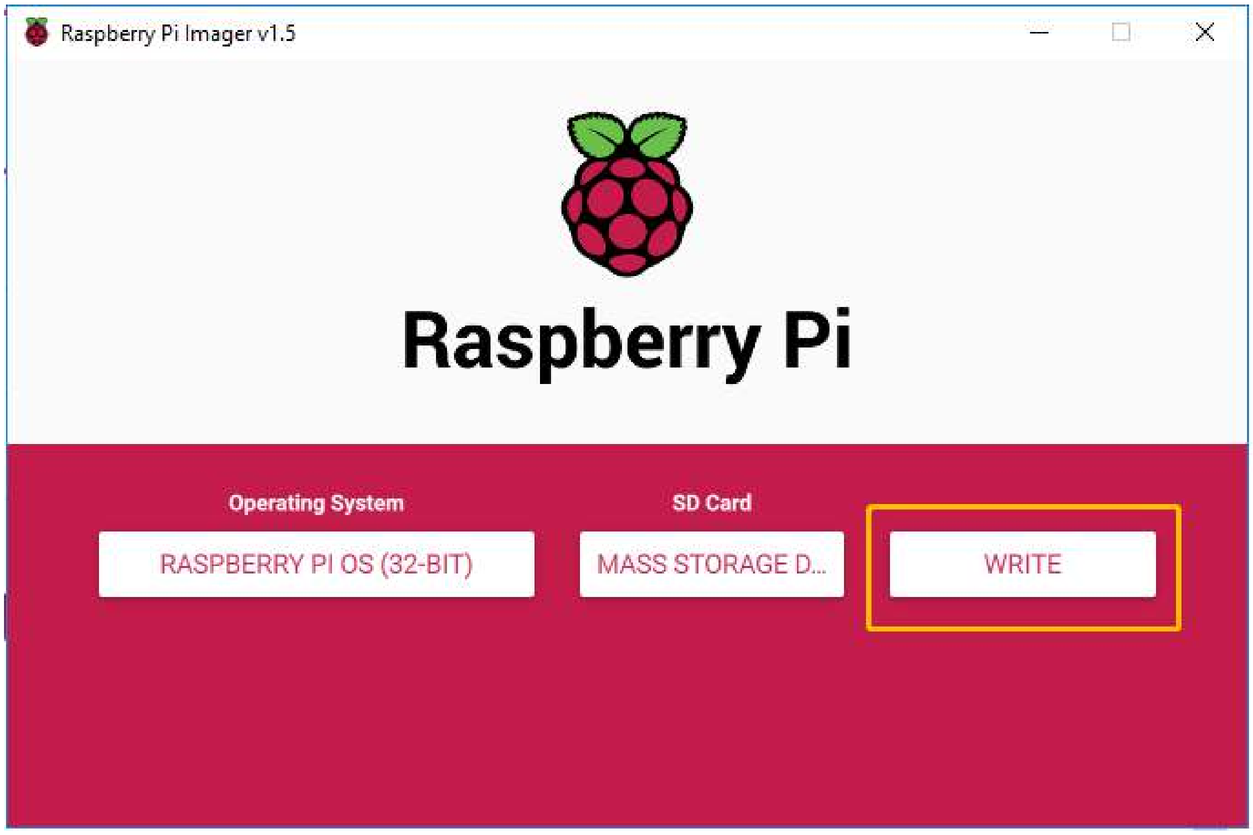
Wifi country: GB

Set locale settings [ ]

Time zone: Asia/Shanghai

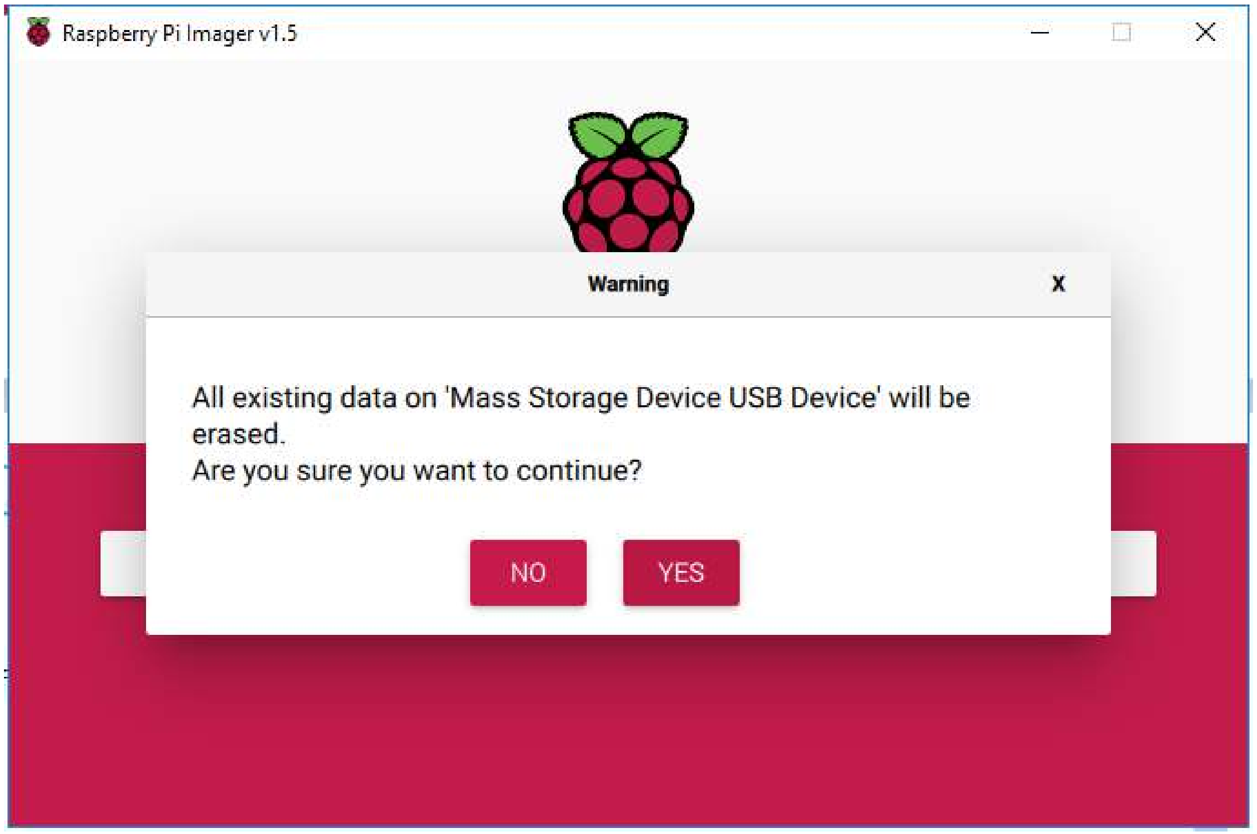
Step 7

Click the **WRITE** button.



Step 8

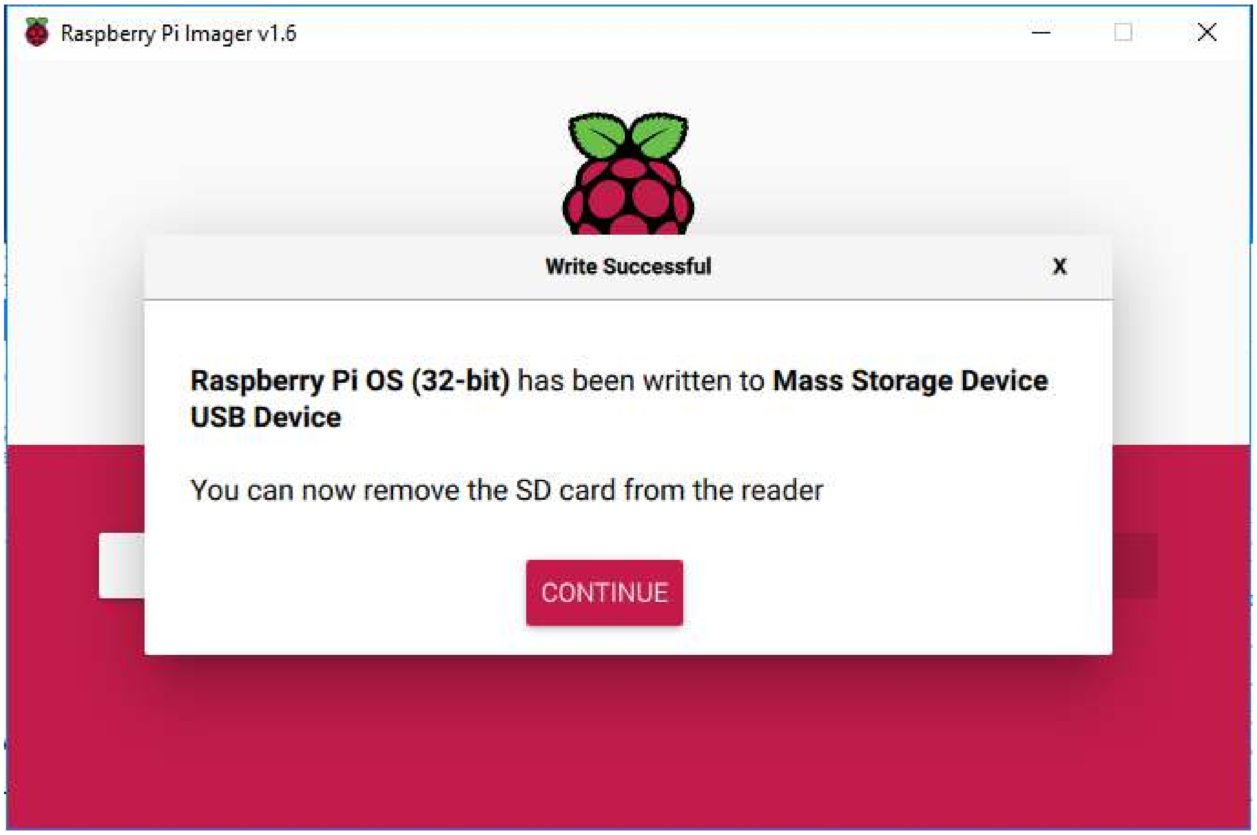
If your SD card currently has any files on it, you may wish to back up these files first to prevent you from permanently losing them. If there is no file to be backed up, click **Yes**.



All existing data on 'Mass Storage Device USB Device' will be erased.
Are you sure you want to continue?

Step 9

After waiting for a period of time, the following window will appear to represent the completion of writing.



Raspberry Pi OS (32-bit) has been written to **Mass Storage Device USB Device**

You can now remove the SD card from the reader

4.1.3 Set up Your Raspberry Pi

If You Have a Screen

If you have a screen, it will be easy for you to operate on the Raspberry Pi.

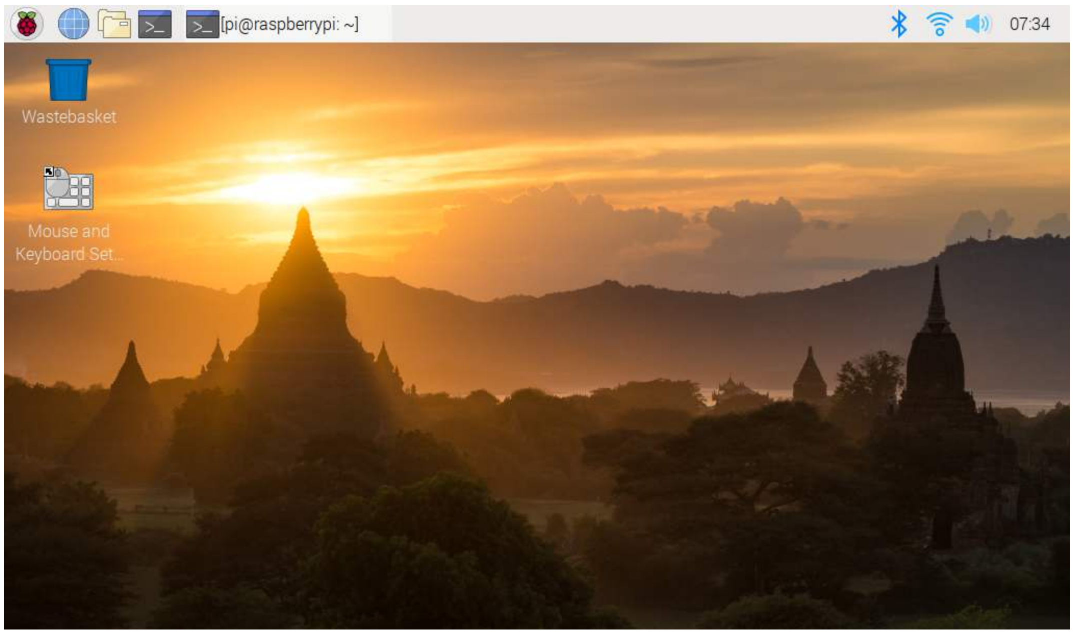
Required Components

Any Raspberry Pi	1 * Power Adapter
1 * Micro SD card	1 * Screen Power Adapter
1 * HDMI cable	1 * Screen
1 * Mouse	1 * Keyboard

1. Insert the SD card you've set up with Raspberry Pi OS into the micro SD card slot on the underside of your Raspberry Pi.
2. Plug in the Mouse and Keyboard.
3. Connect the screen to Raspberry Pi's HDMI port and make sure your screen is plugged into a wall socket and switched on.

   Note: If you use a Raspberry Pi 4, you need to connect the screen to the HDMI0 (nearest the power in port).

4. Use the power adapter to power the Raspberry Pi. After a few seconds, the Raspberry Pi OS desktop will be displayed.



If You Have No Screen

If you don't have a display, you can log in to the Raspberry Pi remotely, but before that, you need to get the IP of the Raspberry Pi.

Get the IP Address

After the Raspberry Pi is connected to WIFI, we need to get the IP address of it. There are many ways to know the IP address, and two of them are listed as follows.

1. Checking via the router

If you have permission to log in the router(such as a home network), you can check the addresses assigned to Raspberry Pi on the admin interface of router.

The default hostname of the Raspberry Pi OS is **raspberrypi**, and you need to find it. (If you are using ArchLinuxARM system, please find alarmpi.)

2. Network Segment Scanning

You can also use network scanning to look up the IP address of Raspberry Pi. You can apply the software, **Advanced IP scanner** and so on.

Scan the IP range set, and the name of all connected devices will be displayed. Similarly, the default hostname of the Raspberry Pi OS is **raspberrypi**, if you haven't modified it.

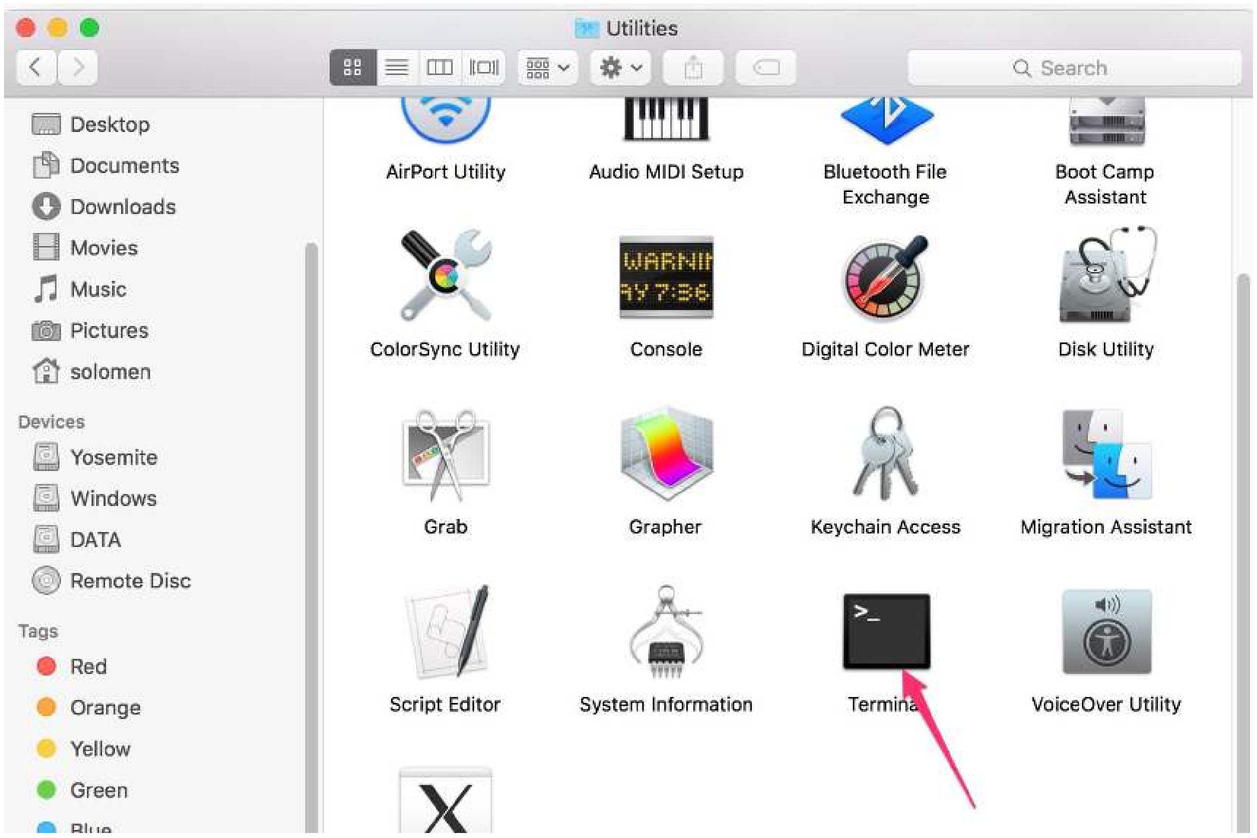
Use the SSH Remote Control

We can open the Bash Shell of Raspberry Pi by applying SSH. Bash is the standard default shell of Linux. The Shell itself is a program written in C that is the bridge linking the customers and Unix/Linux. Moreover, it can help to complete most of the work needed.

For Linux or/Mac OS X Users

Step 1

Go to **Applications->Utilities**, find the **Terminal**, and open it.



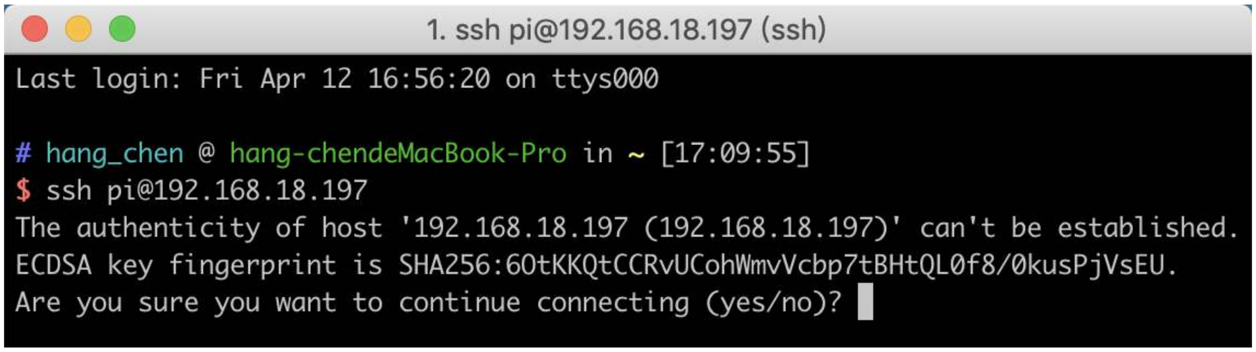
Step 2

Type in **ssh pi@ip_address** . “pi” is your username and “ip_address” is your IP address. For example:

```
ssh pi@192.168.18.197
```

Step 3

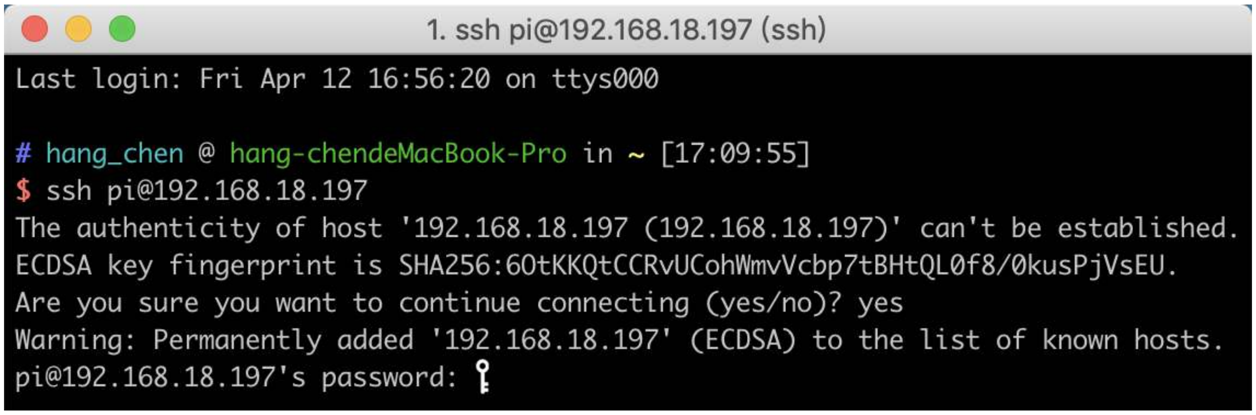
Input “yes”.



```
Last login: Fri Apr 12 16:56:20 on ttys000

# hang_chen @ hang-chendeMacBook-Pro in ~ [17:09:55]
$ ssh pi@192.168.18.197
The authenticity of host '192.168.18.197 (192.168.18.197)' can't be established.
ECDSA key fingerprint is SHA256:6OtKKQtCCRvUCohWmvVcbp7tBHtQL0f8/0kusPjVsEU.
Are you sure you want to continue connecting (yes/no)?
```

Step 4

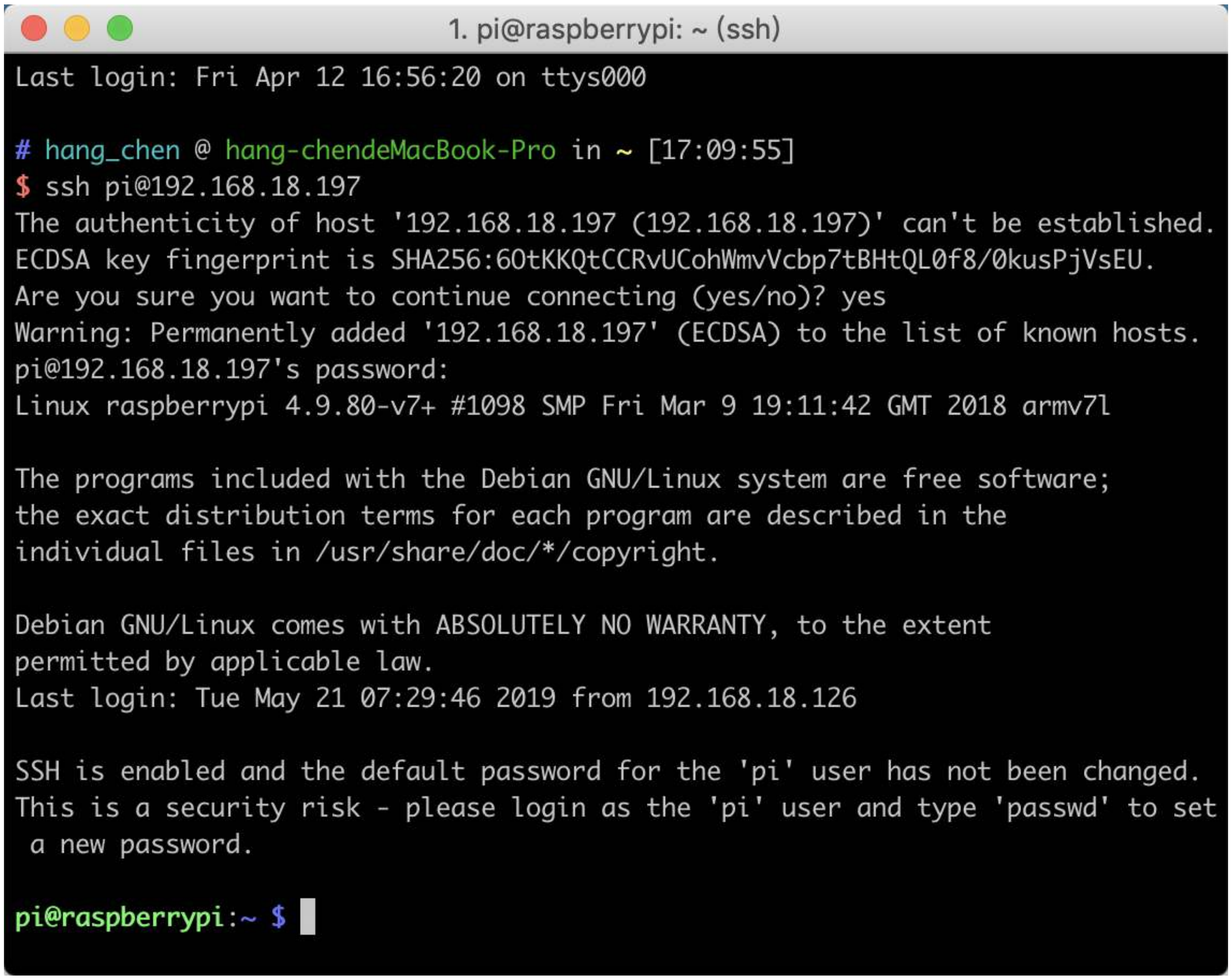
Input the passcode and the default password is **raspberry**.



```
Last login: Fri Apr 12 16:56:20 on ttys000

# hang_chen @ hang-chendeMacBook-Pro in ~ [17:09:55]
$ ssh pi@192.168.18.197
The authenticity of host '192.168.18.197 (192.168.18.197)' can't be established.
ECDSA key fingerprint is SHA256:6OtKKQtCCRvUCohWmvVcbp7tBHtQL0f8/0kusPjVsEU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.18.197' (ECDSA) to the list of known hosts.
pi@192.168.18.197's password:
```

Step 5

We now get the Raspberry Pi connected and are ready to go to the next step.



```
Last login: Fri Apr 12 16:56:20 on ttys000

# hang_chen @ hang-chendeMacBook-Pro in ~ [17:09:55]
$ ssh pi@192.168.18.197
The authenticity of host '192.168.18.197 (192.168.18.197)' can't be established.
ECDSA key fingerprint is SHA256:6OtKKQtCCRvUCohWmvVcbp7tBHtQL0f8/0kusPjVsEU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.18.197' (ECDSA) to the list of known hosts.
pi@192.168.18.197's password:
Linux raspberrypi 4.9.80-v7+ #1098 SMP Fri Mar 9 19:11:42 GMT 2018 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue May 21 07:29:46 2019 from 192.168.18.126

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $
```

Note: When you input the password, the characters do not display on window accordingly, which is normal. What you need is to input the correct password.

For Windows Users

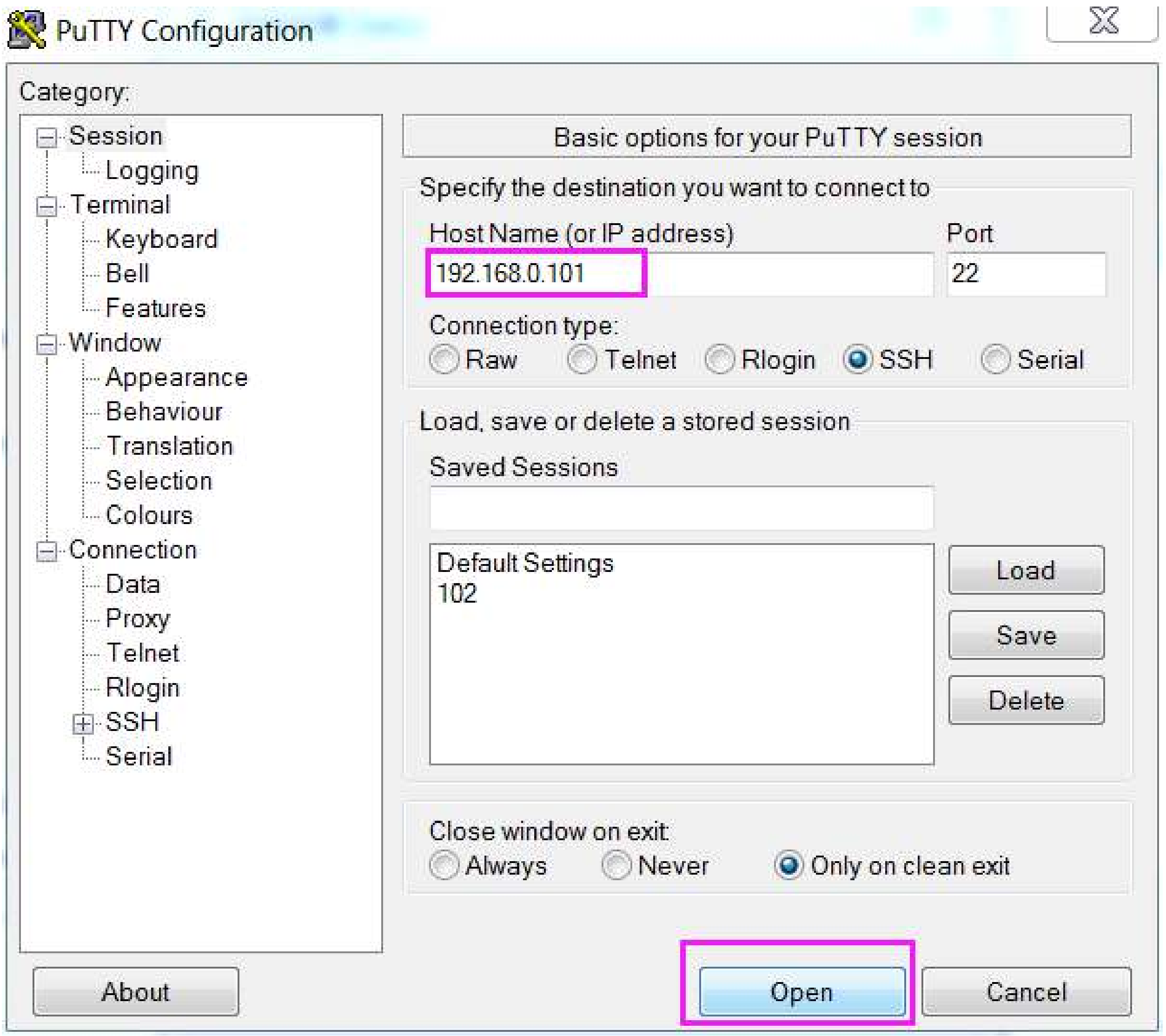
If you're a Windows user, you can use SSH with the application of some software. Here, we recommend **PuTTY**.

Step 1

Download PuTTY.

Step 2

Open PuTTY and click **Session** on the left tree-alike structure. Enter the IP address of the RPi in the text box under **Host Name (or IP address)** and **22** under **Port** (by default it is 22).



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.0.101

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
102

Load Save Delete

Close window on exit: Always Never [✓ Only on clean exit]

Open Cancel

Step 3

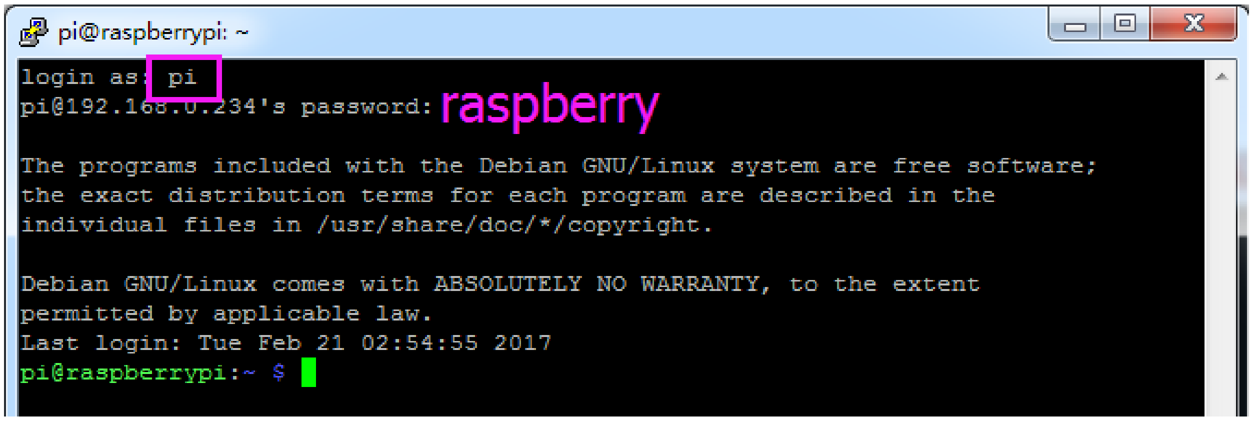
Click **Open**. Note that when you first log in to the Raspberry Pi with the IP address, there prompts a security reminder. Just click **Yes**.

Step 4

When the PuTTY window prompts "**login as:**", type in "**pi**" (the user name of the RPi), and **password:** "raspberry" (the default one, if you haven't changed it).

Note: When you input the password, the characters do not display on window accordingly, which is normal. What you need is to input the correct password.

If inactive appears next to PuTTY, it means that the connection has been broken and needs to be reconnected.



```
login as: pi
pi@192.168.0.234's password: raspberry

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Feb 21 02:54:55 2017
pi@raspberrypi:~ $
```

Step 5

Here, we get the Raspberry Pi connected and it is time to conduct the next steps.

Note: If you are not satisfied with using the command window to control the Raspberry Pi, you can also use the remote desktop function, which can help us manage the files in the Raspberry Pi easily.

For details on how to do this, please refer to *Remote Desktop*.

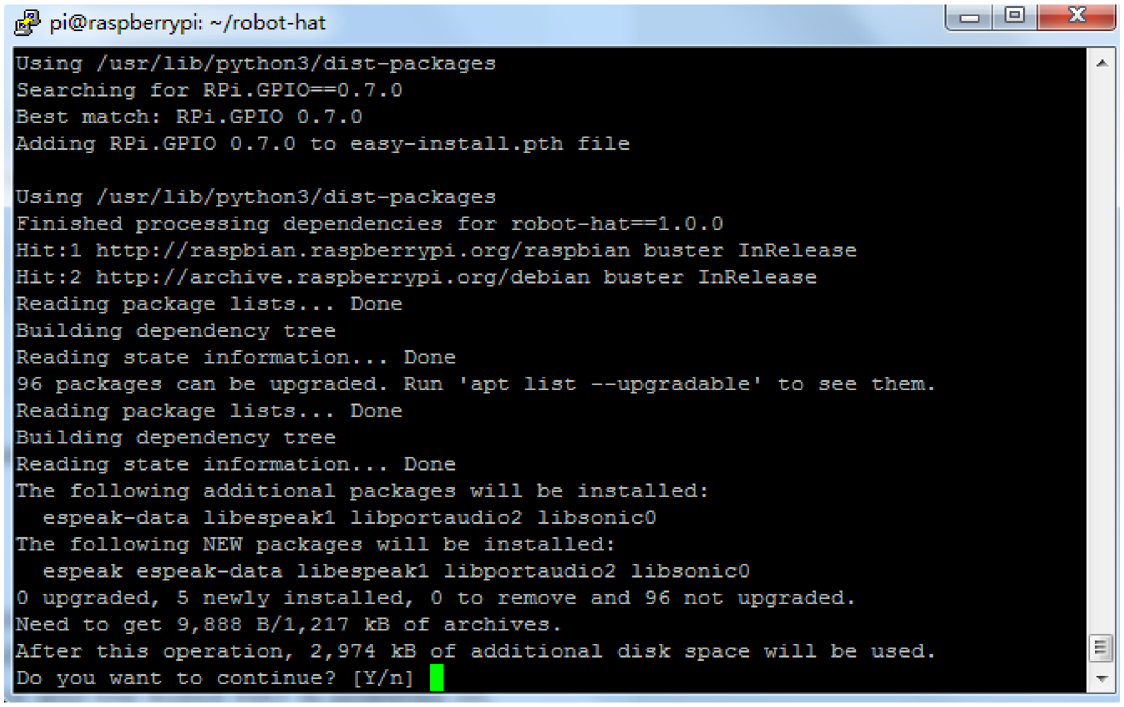
4.1.4 Download and Run the Code

We can download the files by using `git clone` in the command line.

Install `robot-hat` library first.

```
cd /home/pi/
git clone https://github.com/sunfounder/robot-hat.git
cd robot-hat
sudo python3 setup.py install
```

Note: Running setup.py will download some necessary components. You may fail to download due to network problems. You may need to download again at this time. In the following cases, enter Y and press Enter.



```
Using /usr/lib/python3/dist-packages
Searching for RPi.GPIO==0.7.0
Best match: RPi.GPIO 0.7.0
Adding RPi.GPIO 0.7.0 to easy-install.pth file

Using /usr/lib/python3/dist-packages
Finished processing dependencies for robot-hat==1.0.0
Hit:1 http://raspbian.raspberrypi.org/raspbian buster InRelease
Hit:2 http://archive.raspberrypi.org/debian buster InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
96 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  espeak-data libespeak1 libportaudio2 libsonic0
The following NEW packages will be installed:
  espeak espeak-data libespeak1 libportaudio2 libsonic0
0 upgraded, 5 newly installed, 0 to remove and 96 not upgraded.
Need to get 9,888 B/1,217 kB of archives.
After this operation, 2,974 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

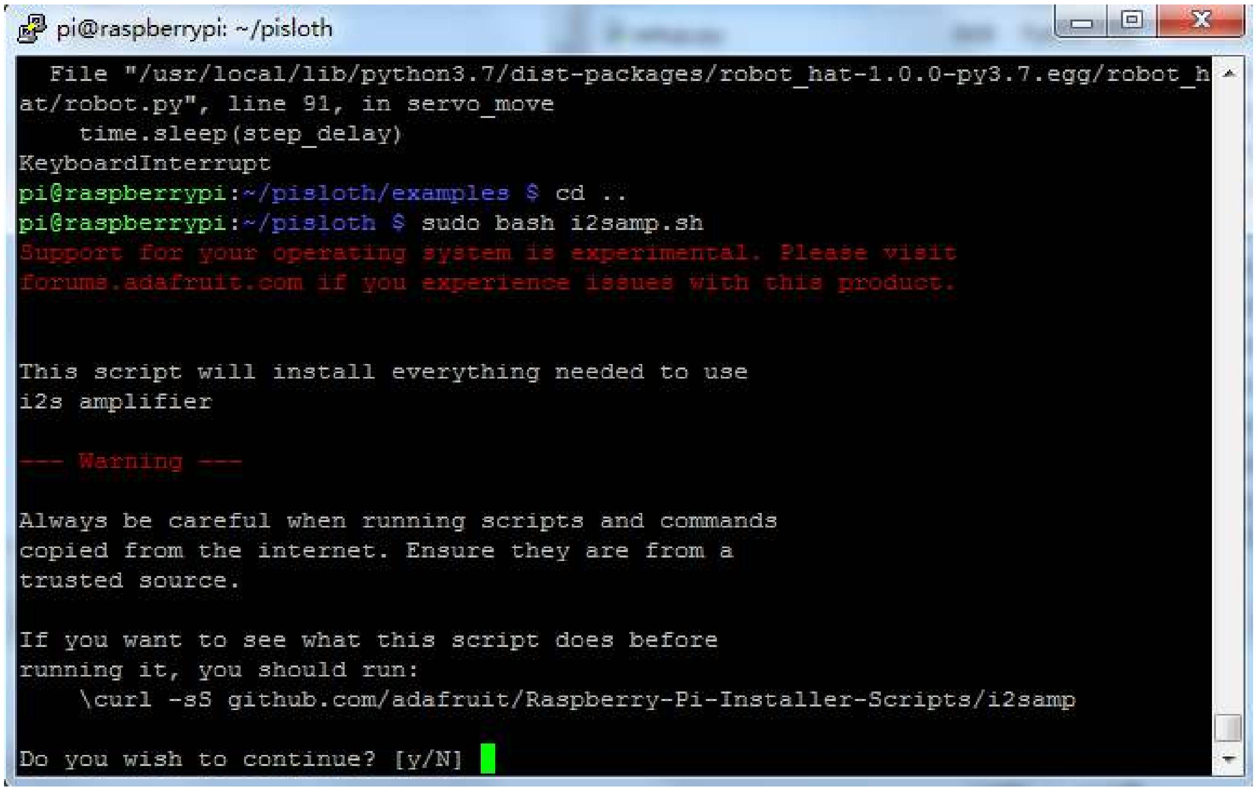
Then download the code and install `pisloth` library.

```
cd /home/pi/
git clone -b v2.0 https://github.com/sunfounder/pisloth.git
cd pisloth
sudo python3 setup.py install
```

This step will take a little time, so please be patient.

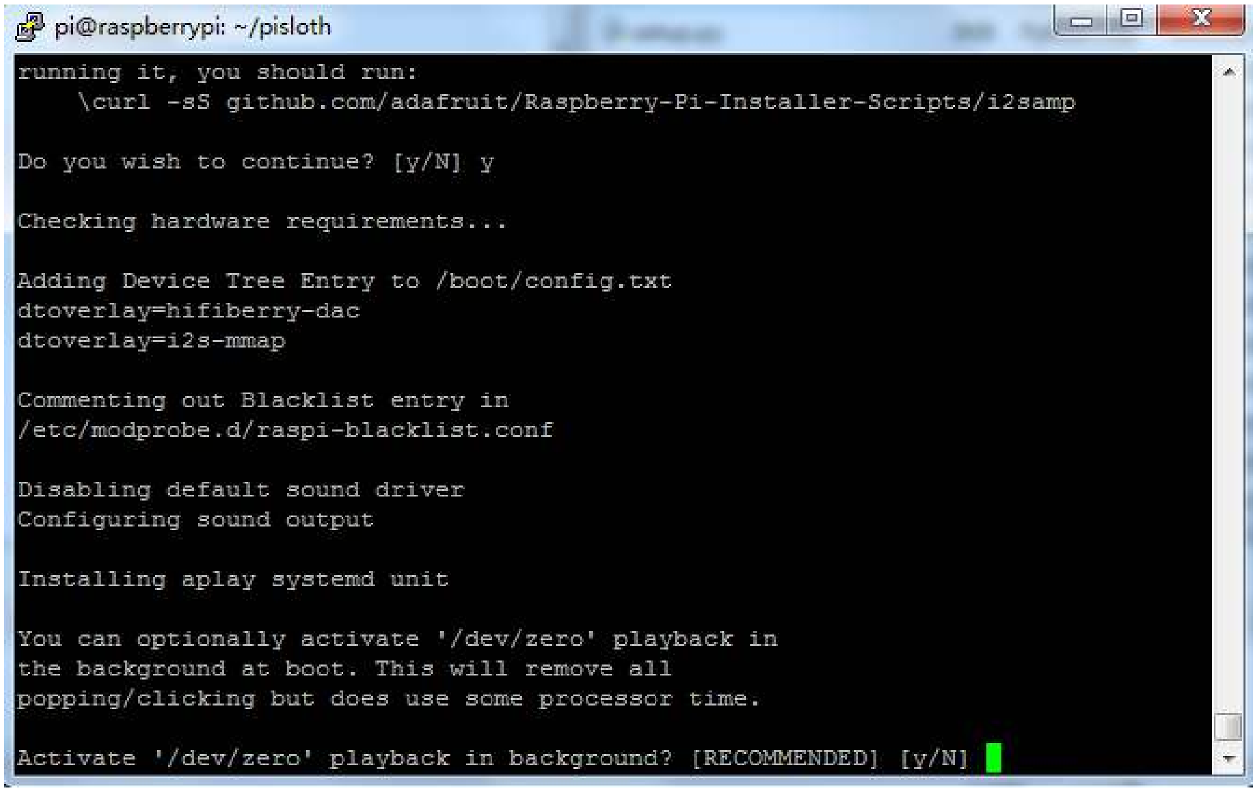
Finally, you need to run the script `i2samp.sh` to install the components required by the i2s amplifier, otherwise the pislot will have no sound.

```
cd /home/pi/pisloth
sudo bash i2samp.sh
```



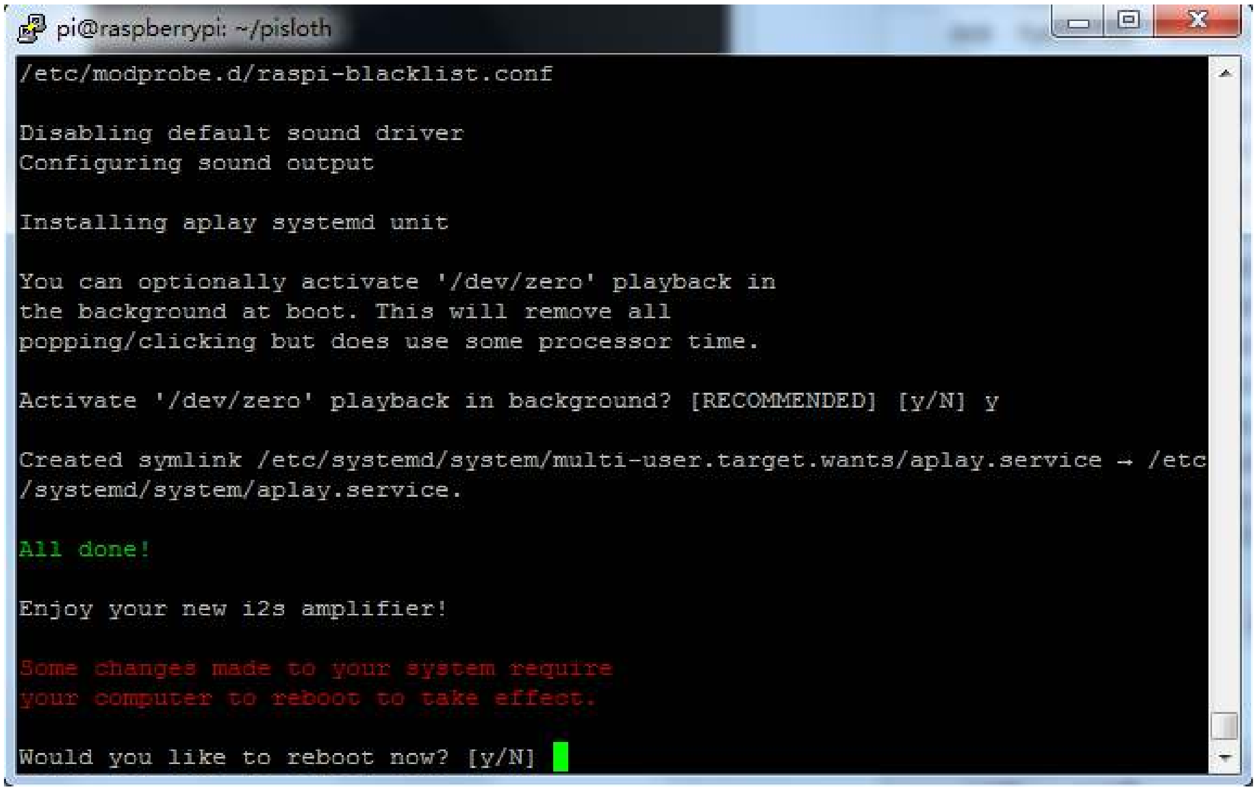
```
  File "/usr/local/lib/python3.7/dist-packages/robot_hat-1.0.0-py3.7.egg/robot_h
at/robot.py", line 91, in servo_move
    time.sleep(step_delay)
KeyboardInterrupt
pi@raspberrypi:~/pisloth/examples $ cd ..
pi@raspberrypi:~/pisloth $ sudo bash i2samp.sh
Support for your operating system is experimental. Please visit
forums.adafruit.com if you experience issues with this product.


This script will install everything needed to use
i2s amplifier

--- Warning ---

Always be careful when running scripts and commands
copied from the internet. Ensure they are from a
trusted source.

If you want to see what this script does before
running it, you should run:
    \curl -sS github.com/adafruit/Raspberry-Pi-Installer-Scripts/i2samp

Do you wish to continue? [y/N]
```

Type y and press Enter to continue running the script.



```
running it, you should run:
    \curl -sS github.com/adafruit/Raspberry-Pi-Installer-Scripts/i2samp

Do you wish to continue? [y/N] y

Checking hardware requirements...

Adding Device Tree Entry to /boot/config.txt
dtoverlay=hifiberry-dac
dtoverlay=i2s-mmap

Commenting out Blacklist entry in
/etc/modprobe.d/raspi-blacklist.conf

Disabling default sound driver
Configuring sound output

Installing aplay systemd unit

You can optionally activate '/dev/zero' playback in
the background at boot. This will remove all
popping/clicking but does use some processor time.

Activate '/dev/zero' playback in background? [RECOMMENDED] [y/N]
```

Type y and press Enter to run `/dev/zero` in the background.



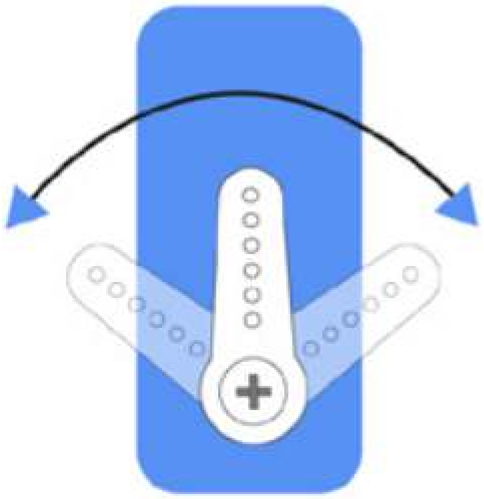
```
pi@raspberrypi: ~/pisloth
/etc/modprobe.d/raspi-blacklist.conf

Disabling default sound driver
Configuring sound output

Installing aplay systemd unit

You can optionally activate '/dev/zero' playback in
the background at boot. This will remove all
popping/clicking but does use some processor time.

Activate '/dev/zero' playback in background? [RECOMMENDED] [y/N] y

Created symlink /etc/systemd/system/multi-user.target.wants/aplay.service → /etc
/systemd/system/aplay.service.

All done!

Enjoy your new i2s amplifier!

Some changes made to your system require
your computer to reboot to take effect.

Would you like to reboot now? [y/N]
```

Type y and press Enter to restart the machine.

Note: If there is no sound after restarting, you may need to run the i2samp.sh script multiple times.

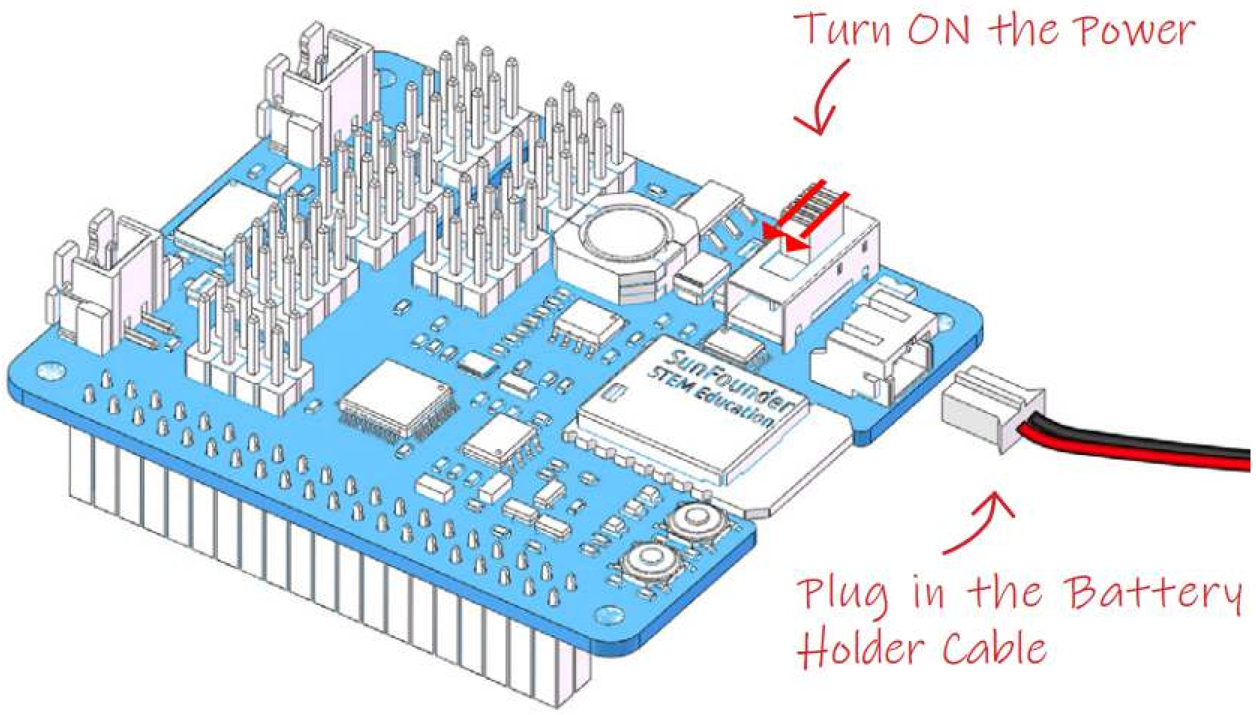
4.1.5 Servo Adjust

To ensure that the servo has been properly set to 0°, first insert the rocker arm into the servo shaft and then gently rotate the rocker arm to a different angle.



Follow the instructions on the assembly foldout, insert the battery holder cable and turn the power switch to the ON.

Wait for 1-2 minutes, there will be a sound to indicate that the Raspberry Pi boots successfully.

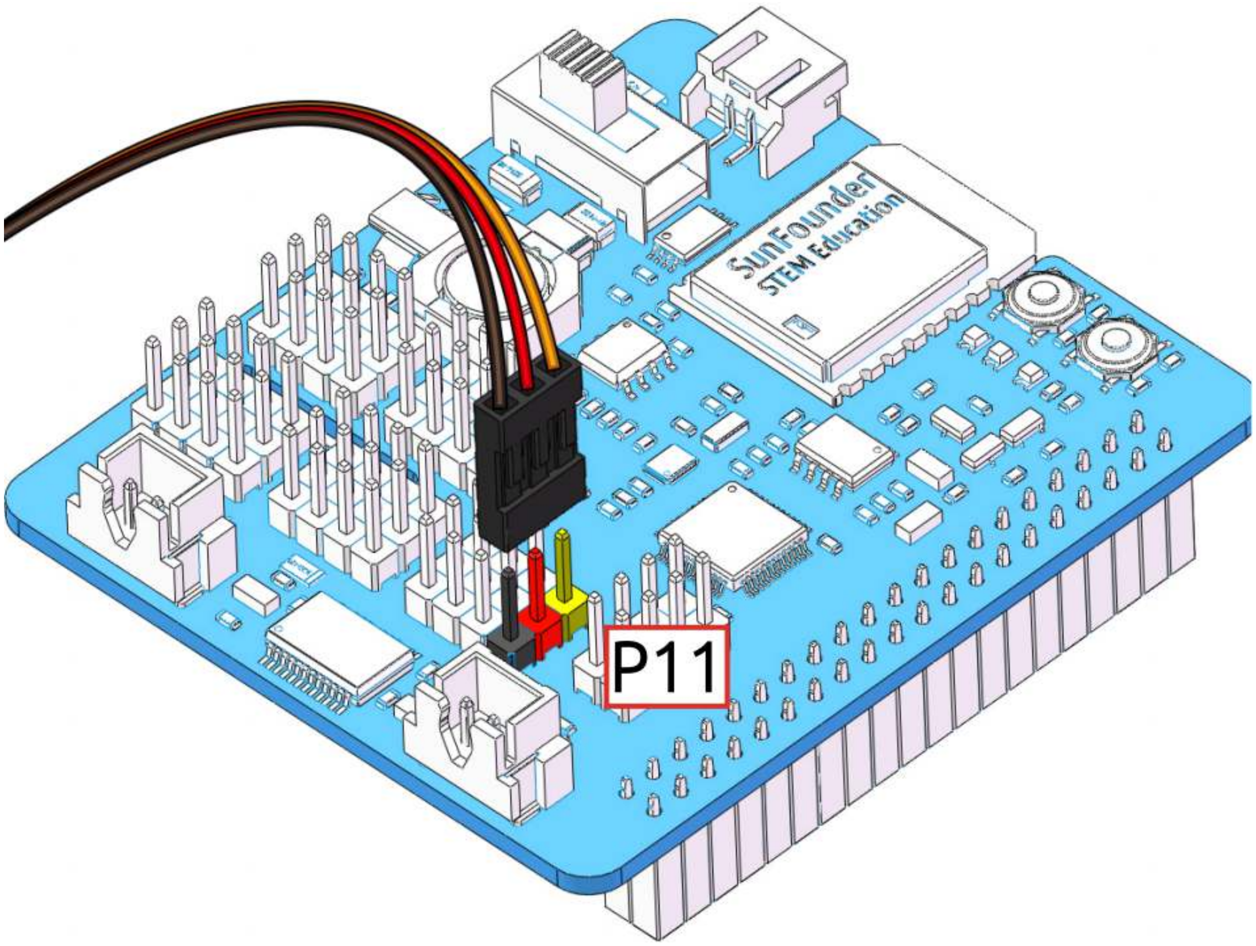


Now, run `servo_zeroing.py` in the `examples/` folder.

```
cd /home/pi/piarm/examples
sudo python3 servo_zeroing.py
```

Note: If you get an error, try re-enabling the Raspberry Pi's I2C port, see: *I2C Configuration*.

Next, plug the servo cable into the P11 port as follows.



At this point you will see the servo arm rotate to a specific position (0°). If the servo arm does not return to 0°, press the RST button to restart the Robot HAT.

Now you can continue the installation as instructed on the assembly foldout.

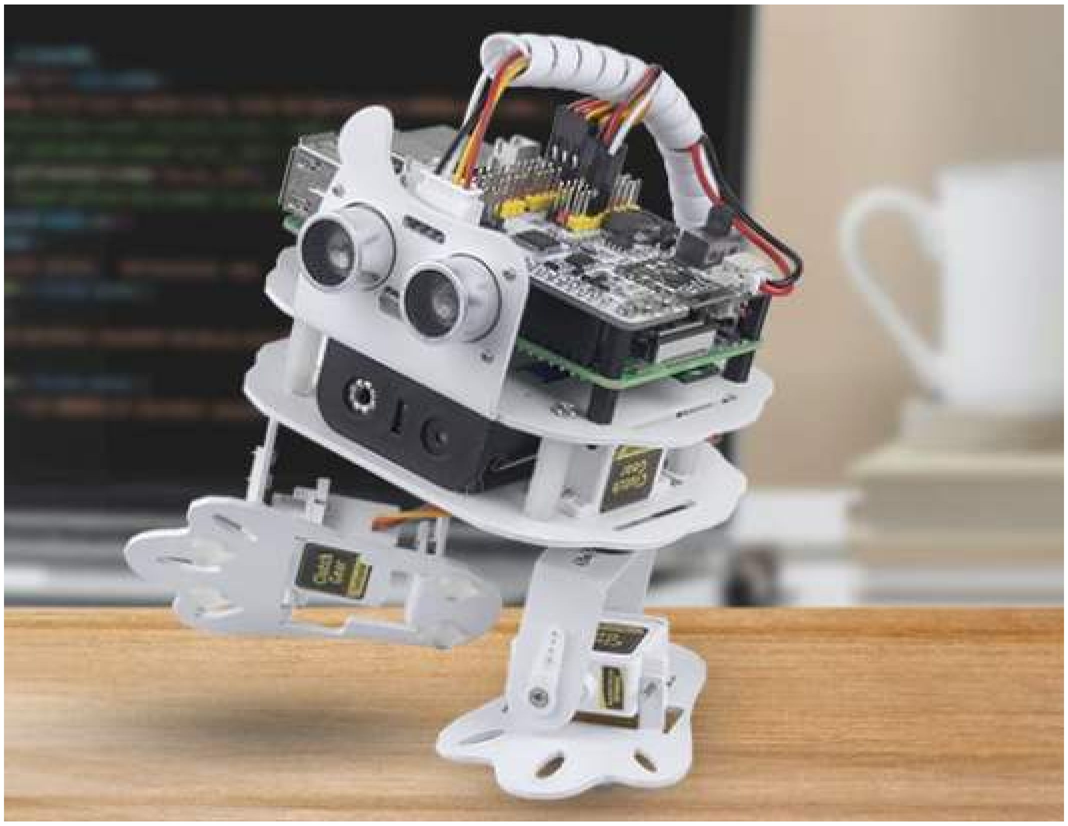
Note:

- Do not unplug this servo cable before fixing it with the servo screw, you can unplug it after fixing it.
- Do not rotate the servo while it is powered on to avoid damage; if the servo shaft is not inserted at the right angle, pull the servo out and reinsert it.
- Before assembling each servo, you need to plug the servo cable into P11 and turn on the power to set its angle to 0°.

After the assembly is complete, you can try to run the projects below.

4.2 Move

This is the first project. PiSloth has woken up, and it moves freely.



Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 move.py
```

After running the code, you will see PiSloth move left 7 steps, forward 5 steps, right 7 steps, and forward 5 steps.

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth

sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])

def main():
    sloth.do_action('turn left', 7, 90)
    sloth.do_action('forward', 5, 90)
    sloth.do_action('turn right', 7, 90)
    sloth.do_action('forward', 5, 90)

```

(continues on next page)

(continued from previous page)

```
if __name__ == "__main__":
    while True:
        main()
```

How it works?

First, import the `Sloth` class from the `pisloth` library you have installed, which contains all of PiSloth's actions and the functions that implement them.

```
from pisloth import Sloth
```

Then instantiate the `Sloth` class.

```
sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])
```

Finally use the `sloth.do_action()` function to make PiSloth move.

```
sloth.do_action('turn left', 7, 90)
sloth.do_action('forward', 5, 90)
sloth.do_action('turn right', 7, 90)
sloth.do_action('forward', 5, 90)
```

In general, all actions of PiSloth can be implemented with the `sloth.do_action()` function. It has four parameters:

- `motion_name` is the name of specific actions, including: `forward`, `turn right`, `turn left`, `backward`, `stand`, `moon walk left`, `moon walk right`, `hook`, `big swing`, `swing`, `walk boldly`, `walk backward boldly`, `walk shyly`, `walk backward shyly`, `stomp rihgt`, `stomp left`, `close`, `open`, `tiptoe left`, `tiptoe right`, `fall left`, `fall right`.
- `step` represents the number of each action is done, the default is 1.
- `speed` indicates the speed of the action, the default is 50 and the range is 0~100.
- `bpm` means rhythm, we will use it later in the *Dance* project.

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.3 Dance

Now, PiSltoh will show you its newly learned dance.

Note: You can download and print the PDF Cartoon Mask for your PiSloth.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 dancing.py
```

The whole dance is divided into 2 parts, and PiSloth will finish these 2 parts with the music. If you don't stop the code, it will repeat the dance.

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import Music
from robot_hat import Ultrasonic
from robot_hat import Pin
import time
import os

music = Music()

sloth = Sloth([1,2,3,4])
```

(continues on next page)

(continued from previous page)

```
sloth.set_offset([0,0,0,0])


def main():

    music.background_music('./musics/india-Arulo.mp3')
    music.music_set_volume(20)
    sloth.do_action('stomp left',3,bpm=129)
    sloth.do_action('stomp right',3,bpm=129)
    sloth.do_action('moon walk left',3,bpm=129)
    sloth.do_action('moon walk right',3,bpm=129)
    for i in range(3):
        sloth.do_action('swing',1,bpm=129)
        sloth.do_action('stand',1,bpm=129)
    for i in range(3):
        sloth.do_action('close',1,bpm=129)
        sloth.do_action('stand',1,bpm=129)
        sloth.do_action('open',1,bpm=129)
        sloth.do_action('stand',1,bpm=129)
    sloth.do_action('tiptoe left',2,bpm=129)
    sloth.do_action('tiptoe right',2,bpm=129)

    sloth.do_action('stomp left',3,bpm=129)
    sloth.do_action('stomp rihgt',3,bpm=129)
    sloth.do_action('moon walk left',3,bpm=129)
    sloth.do_action('moon walk right',3,bpm=129)
    for i in range(3):
        sloth.do_action('hook',1,bpm=129)
        sloth.do_action('stand',1,bpm=129)
    for i in range(4):
        sloth.do_action('swing',1,bpm=129)
        sloth.do_action('big swing',1,bpm=129)
        sloth.do_action('swing',1,bpm=129)
        sloth.do_action('stand',1,bpm=129)

    sloth.do_action('tiptoe right',2,bpm=129)
    sloth.do_action('stand',2,bpm=129)



    music.music_stop()
    time.sleep(10)



if __name__ == "__main__":
    while True:
        main()
```

How it works?

You can make PiSloth play music by importing the following libraries.

```
from robot_hat import TTS, Music
```

Play the background music in the `pisloth/examples/musics` directory and set the volume to 20. You can also add music to the `musics` folder via *Filezilla Software*.

```
music.background_music('./musics/india-Arulo.mp3')
music.music_set_volume(20)
```

In general, all actions of PiSloth can be implemented with the `sloth.do_action()` function. It has four parameters:

- `motion_name` is the name of specific actions, including: `forward`, `turn right`, `turn left`, `backward`, `stand`, `moon walk left`, `moon walk right`, `hook`, `big swing`, `swing`, `walk boldly`, `walk backward boldly`, `walk shyly`, `walk backward shyly`, `stomp rihgt`, `stomp left`, `close`, `open`, `tiptoe left`, `tiptoe right`, `fall left`, `fall right`.
- `step` represents the number of each action is done, the default is 1.
- `speed` indicates the speed of the action, the default is 50 and the range is 0~100.
- `bpm` means rhythm, the bpm parameter here affects the interval time of PiSloth movement. The higher the value, the shorter the interval time. When we know the beat of a song through the **bpm calculator**, we can make PiSloth dance to the music.

For music bmp, if you want to know more, you can refer to: https://en.wikipedia.org/wiki/Tempo

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.4 Obstacle Avoidance

In this project, PiSloth will use an ultrasonic module to detect obstacles in front. When PiSloth detects an obstacle, it will send a signal and look for another direction to move forward.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 avoid.py
```

After the code runs, PiSloth will walk forward. If it detects that the distance of the obstacle ahead is less than 10cm, it will stop and sound a warning, then turn left and stop. If there is no obstacle in the direction after turning left or the obstacle distance is greater than 10, it will continue to move forward.

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import TTS, Music
from robot_hat import Ultrasonic
from robot_hat import Pin
import time
import os

tts = TTS()
music = Music()

sloth = Sloth([1,2,3,4])
```

(continues on next page)

(continued from previous page)

```
sloth.set_offset([0,0,0,0])
sonar = Ultrasonic(Pin("D2") ,Pin("D3"))

alert_distance = 10

def main():
    distance = sonar.read()
    if distance < 0:
        pass
    elif distance <= alert_distance:
        try:
            music.sound_effect_threading('./sounds/sign.wav')
        except Exception as e:
            print(e)
        sloth.do_action('hook', 1,95)
        time.sleep(0.5)
        sloth.do_action('stand', 1,95)
        time.sleep(0.5)
        sloth.do_action('turn left',7,90)
        sloth.do_action('stand', 1,95)
        time.sleep(0.2)
    else :
        sloth.do_action('forward', 1,90)


if __name__ == "__main__":
    while True:
        main()
```

How it works?

You can get the distance by importing the `Ultrasonic` class.

```
from robot_hat import Ultrasonic
```

Then initialize the ultrasonic pins.

```
sonar = Ultrasonic(Pin("D2") ,Pin("D3"))
```

Here is the main program.

- Read the `distance` detected by ultrasonic module and filter out the values less than 0 (When the ultrasonic module is too far from the obstacle or cannot read the data correctly, `distance<0` will appear).
- When the `distance` is less than or equal to `alert_distance` (the threshold value set earlier, which is 10), play the sound effect `sign.wav`. PiSloth does `hook`, `stand`, `left turn` and `stand` in sequence.
- When the `distance` is greater than `alert_distance`, PiSloth will move `forward`.

```
distance = sonar.read()
if distance < 0:
    pass
elif distance <= alert_distance:
    try:
        music.sound_effect_threading('./sounds/sign.wav')
    except Exception as e:
        print(e)
    sloth.do_action('hook', 1,95)
```

(continues on next page)

(continued from previous page)

```
    time.sleep(0.5)
    sloth.do_action('stand', 1,95)
    time.sleep(0.5)
    sloth.do_action('turn left',7,90)
    sloth.do_action('stand', 1,95)
    time.sleep(0.2)
else :
    sloth.do_action('forward', 1,90)
```

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.5 Don't Touch Me

If you don't meet PiSloth's needs, it will get angry and stay away from your touch.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 dont_touch_me.py
```

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import Music
from robot_hat import Ultrasonic
from robot_hat import Pin
import time
import os


music = Music()

sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])
sonar = Ultrasonic(Pin("D2") ,Pin("D3"))

alert_distance = 20

def main():
    distance = sonar.read()
    print(distance)
    if distance <= alert_distance :
        try:
            music.sound_effect_threading('./sounds/talk3.wav')
        except Exception as e:
            print(e)
        sloth.do_action('backward', 2, 90)
    else:
```

(continues on next page)

(continued from previous page)

```
        sloth.do_action('stand', 1, 90)
        time.sleep(1)


if __name__ == "__main__":
    while True:
        main()
```

How it works?

Instantiate various classes of `Music`, `Sloth` and `Ultrasonic` to be used.

```
music = Music()

sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])
sonar = Ultrasonic(Pin("D2") ,Pin("D3"))
```

Here is the main program.

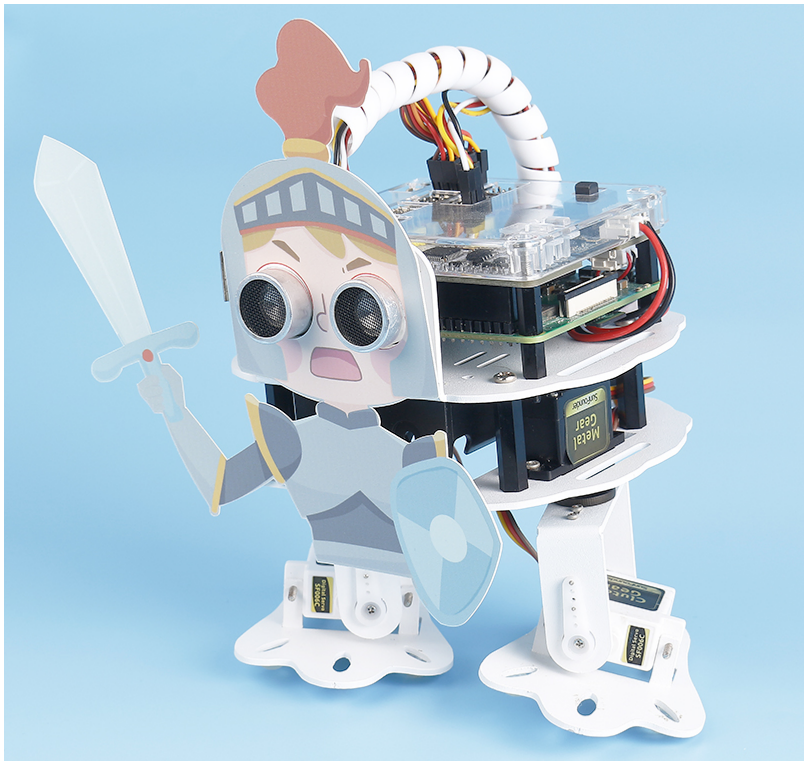
- Read the `distance` detected by the ultrasonic module and print it.
- When the `distance` is less than or equal to `alert_distance` (the threshold value set earlier, which is 20), play the sound effect `talk3.wav` and move `backward`.
- When the `distance` is greater than `alert_distance`, PiSloth will Stand.

```
distance = sonar.read()
print(distance)
if distance <= alert_distance :
    try:
        music.sound_effect_threading('./sounds/talk3.wav')
    except Exception as e:
        print(e)
    sloth.do_action('backward', 2, 90)
else:
    sloth.do_action('stand', 1, 90)
    time.sleep(1)
```

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.6 Let's Fight! Warrior!

Here, PiSloth is a brave warrior, when it appears in front of the enemy, it will let out a roar and rush to the enemy.



Note: You can download and print the PDF Cartoon Mask for your PiSloth.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 lets_fight.py
```

After the code is run, PiSloth will continuously detect the distance of the obstacle, when the distance is between 5 and 40, PiSloth will make a roaring sound and rush forward; when the distance of the obstacle is less than 5, PiSloth will stop.

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import Music
from robot_hat import Ultrasonic
from robot_hat import Pin
import time
import os

```

(continues on next page)

(continued from previous page)

```python

music = Music()

sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])
sonar = Ultrasonic(Pin("D2") ,Pin("D3"))

alert_distance = 40
contact_distance = 5

def main():
    distance = sonar.read()
    if distance <= alert_distance and distance >= contact_distance :
        try:
            music.sound_effect_play('./sounds/battle.wav')
            music.background_music('./musics/attack.mp3')
            music.music_set_volume(20)
        except Exception as e:
            print(e)
        while True:
            distance = sonar.read()
            print(distance)
            if distance < 0:
                continue
            if distance <= contact_distance:
                break
            sloth.do_action('forward', 1,90)
    sloth.do_action('stand', 1, 90)
    time.sleep(1)


if __name__ == "__main__":
    while True:
        main()
```

How it works?

Here is the main program.

- Read the `distance` detected by ultrasonic module and filter out the values less than 0 (When the ultrasonic module is too far from the obstacle or cannot read the data correctly, `distance<0` will appear).
- When the `distance` is between 5 and 40, PiSloth will play `warning.wav` and `attack.mp3` and move `forward`.
- When the `distance` is less than 5, PiSloth will keep the `stand` position.

```python
distance = sonar.read()
if distance <= alert_distance and distance >= contact_distance :
    try:
        music.sound_effect_play('./sounds/battle.wav')
        music.background_music('./musics/attack.mp3')
        music.music_set_volume(20)
    except Exception as e:
        print(e)
    while True:
        distance = sonar.read()
        print(distance)
```

(continues on next page)

(continued from previous page)

```
        if distance< 0:
            continue
        if distance<=contact_distance:
            break
        sloth.do_action('forward', 1,95)
sloth.do_action('stand', 1, 90)
time.sleep(1)
```

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.7 Emotional PiSloth

PiSloth is very emotional, sometimes happy, sometimes shy, sometimes confused.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 emotional_pisloth.py
```

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import TTS, Music
import time

tts = TTS()
music = Music()

sloth = Sloth([1,2,3,4])
sloth.set_offset([0,0,0,0])

def confuse():
    try:
        music.sound_effect_threading('./sounds/sign.wav')
    except Exception as e:
        print(e)
    sloth.do_action('hook', 1, 90)

def happy():
    try:
        music.sound_effect_threading('./sounds/happy2.wav')
    except Exception as e:
        print(e)
    for i in range(3):
        sloth.do_action('hook', 1, 90)
        sloth.do_action('stand', 1, 90)

def fear():
```

(continues on next page)

(continued from previous page)

```
    try:
        music.sound_effect_threading('./sounds/warning.wav')
    except Exception as e:
        print(e)
    sloth.do_action('hook', 1, 90)
    sloth.do_action('stand', 1, 90)
    try:
        music.sound_effect_threading('./sounds/warning.wav')
    except Exception as e:
        print(e)
    sloth.do_action('walk backward boldly', 1, 90)
    sloth.do_action('stand', 1, 90)

def sad():
    try:
        music.sound_effect_threading('./sounds/depress.wav')
    except Exception as e:
        print(e)
    sloth.do_action('big swing', 1, 90)

def angry():
    try:
        music.sound_effect_threading('./sounds/error.wav')
    except Exception as e:
        print(e)
    sloth.do_action('walk backward boldly', 1, 90)
    sloth.do_action('stand', 1, 90)

def fail():
    try:
        music.sound_effect_threading('./sounds/depress2.wav')
    except Exception as e:
        print(e)
    sloth.do_action('fall left', 1, 90)

def shy():
    try:
        music.sound_effect_threading('./sounds/talk3.wav')
    except Exception as e:
        print(e)
    sloth.do_action('close', 1, 90)
    time.sleep(1)
    try:
        music.sound_effect_threading('./sounds/talk2.wav')
    except Exception as e:
        print(e)
    sloth.do_action('stand', 1, 90)

def main():

    print("shy")
    shy()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("confuse")
```

(continues on next page)

(continued from previous page)

```
    confuse()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("happy")
    happy()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("fear")
    fear()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("sad")
    sad()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("angry")
    angry()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)

    print("fail")
    fail()
    time.sleep(1)
    sloth.do_action('stand', 1, 90)
    time.sleep(2)


if __name__ == "__main__":
    while True:
        main()
```

How it works?

In this project, actions + sound effects are combined into different emotional actions, and you can also modify them yourself.

Note: This `fail` action will make the PiSloth fall, be careful not to let it fall off the table and break it.

You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.8 Remote Control

In this project, we will learn how to use the keyboard to remotely control the PiSloth. You can control the PiSloth to move up, down, left, and right and speak through specific keys.

Run the Code

```
cd /home/pi/pisloth/examples
sudo python3 keyboard_control.py
```

Once the code runs, you can control PiSloth by pressing `wasd`, play different sound effects by pressing `1234`, and make PiSloth talk by pressing `qe`.

Press `esc` to exit.

- w: Go Forward
- a: Turn Left
- s: Backward
- d: Turn Right
- 1: Sound effect: talk1
- 2: Sound effect: talk2
- 3: Sound effect: talk3
- 4: Sound effect: depress2
- q: Say: "Oh hello there"
- e: Say: "bye"
- esc: Quit

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
from robot_hat import Music
from robot_hat import TTS
import sys
import tty
import termios
import time

sloth = Sloth([1,2,3,4])
tts = TTS()
music = Music()
sloth.set_offset([0,0,0,0])

def readchar():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
```

(continues on next page)

(continued from previous page)

```
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    return ch

manual = '''
Press keys on keyboard to control PiSloth!

    w: Forward
    a: Turn left
    s: Backward
    d: Turn right
    1: Sound effect: talk1
    2: Sound effect: talk2
    3: Sound effect: talk3
    4: Sound effect: depress2
    q: Say: "Oh hello there"
    e: Say: "bye"
    esc: Quit
'''

def main():
    print(manual)
    while True:
        key = readchar().lower()
        # print(key)
        if key == "w":
            sloth.do_action('forward', 1, 90)
        elif key == "a":
            sloth.do_action('turn left', 1, 90)
        elif key == "s":
            sloth.do_action('backward', 1, 90)
        elif key == "d":
            sloth.do_action('turn right', 1, 90)
        elif key == "1":
            music.sound_effect_play('./sounds/talk1.wav')
        elif key == "2":
            music.sound_effect_play('./sounds/talk2.wav')
        elif key == "3":
            music.sound_effect_play('./sounds/talk3.wav')
        elif key == "4":
            music.sound_effect_play('./sounds/depress.wav')
        elif key == "q":
            tts.say("Oh hello there")
        elif key == "e":
            tts.say("bye")
        elif key == chr(27): # 27 for ESC
            break
        time.sleep(0.05)
    print("\nQuit")

if __name__ == "__main__":
    main()
```

How it works?

This function refers to the standard input stream and returns the first character of the data stream read.

- `tty.setraw(sys.stdin.fileno)` is to change the standard input stream to raw mode, that is, all characters will not be escaped during transmission, including special characters. Before changing the mode, back up the original mode, and restore it after the change.
- `old_settings = termios.tcgetattr(fd)` and `termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)` plays the role of backup and restore.

```
def readchar():
            fd = sys.stdin.fileno()
            old_settings = termios.tcgetattr(fd)
            try:
                    tty.setraw(sys.stdin.fileno())
                    ch = sys.stdin.read(1)
            finally:
                    termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
            return ch
```

Finally, according to the read keyboard characters, let PiSloth do the actions we set, call the `tts.say()` function to speak or play the sound effects prepared in advance.

```
key = readchar().lower()
    # print(key)
    if key == "w":
        sloth.do_action('forward', 1, 90)
    elif key == "a":
        sloth.do_action('turn left', 1, 90)
    elif key == "s":
        sloth.do_action('backward', 1, 90)
    elif key == "d":
        sloth.do_action('turn right', 1, 90)
    elif key == "1":
        music.sound_effect_play('./sounds/talk1.wav')
    elif key == "2":
        music.sound_effect_play('./sounds/talk2.wav')
    elif key == "3":
        music.sound_effect_play('./sounds/talk3.wav')
    elif key == "4":
        music.sound_effect_play('./sounds/depress.wav')
    elif key == "q":
        tts.say("Oh hello there")
    elif key == "e":
        tts.say("bye")
    elif key == chr(27): # 27 for ESC
        break
```

Note: You can add different sound effects or music to `musics` or `sounds` folder via *Filezilla Software*.

4.9 Custom Step

In the previous projects, we used a lot of actions that we wrote, so how are these actions composed and done? Generally speaking, an action is composed of one or more steps.

In this project, we will learn how to customize PiSloth's step.

Note: You can download and print the PDF Cartoon Mask for your PiSloth.

Run the Code

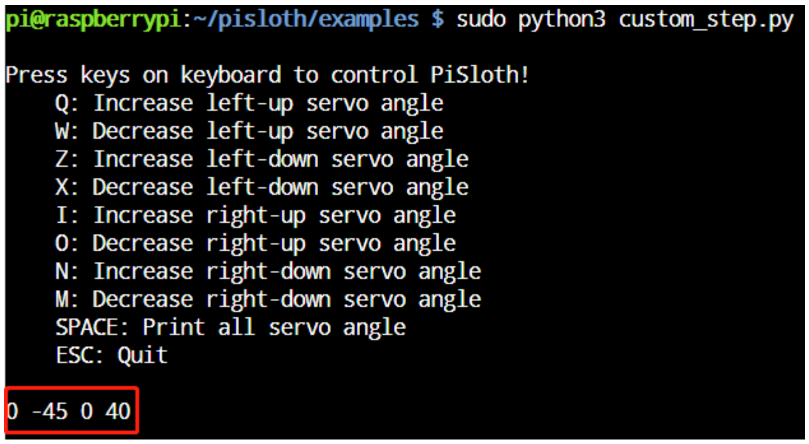
```
cd /home/pi/pisloth/examples
sudo python3 custom_step.py
```

Once the code has been run, press the following keys to adjust the angle of each servo of PiSloth.

- q: Increase the angle of the left leg
- w: Decrease the angle of the left leg
- z: Increase the angle of the left foot
- x: Decreases the angle of the left foot
- i: Increase the angle of the right leg
- o: decreases the angle of the right leg
- n: increases the angle of the right foot
- m: decreases the angle of the right foot
- SPACE: Print all angle
- ESC: exit

For example, by pressing the `zx` and `nm` keys, we make PiSloth do the pose shown in the figure.

Press the **key SPACE** to print the angle of the 4 servos at this time. You need to record these angle values, which will be used in the next project *Custom Action*.



```
pi@raspberrypi:~/pisloth/examples $ sudo python3 custom_step.py

Press keys on keyboard to control PiSloth!
    Q: Increase left-up servo angle
    W: Decrease left-up servo angle
    Z: Increase left-down servo angle
    X: Decrease left-down servo angle
    I: Increase right-up servo angle
    O: Decrease right-up servo angle
    N: Increase right-down servo angle
    M: Decrease right-down servo angle
    SPACE: Print all servo angle
    ESC: Quit

0 -45 0 40
```

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
# from robot_hat import Music
```

(continues on next page)

(continued from previous page)

```
# from robot_hat import TTS
from robot_hat import PWM
from robot_hat import Servo

import sys
import tty
import termios
import time

sloth = Sloth([1,2,3,4])
# tts = TTS()
# music = Music()
sloth.set_offset([0,0,0,0])

right_leg_servo = Servo(PWM('P0'))
right_foot_servo = Servo(PWM('P1'))
left_leg_servo = Servo(PWM('P2'))
left_foot_servo = Servo(PWM('P3'))


def readchar():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    return ch

manual = '''
Press keys on keyboard to control PiSloth!
    q: Increase the servo angle of the left leg
    w: Decrease the servo angle of the left leg
    z: Increase the servo angle of the left foot
    x: Decrease the servo angle of the left foot
    i: Increase the servo angle of the right leg
    o: Decrease the servo angle of the right leg
    n: Increase the servo angle of the right foot
    m: Decrease the servo angle of the right foot
    SPACE: Print all angle
    ESC: Quit
'''

def main():
    print(manual)

    left_leg=0
    left_foot=0
    right_leg=0
    right_foot=0
    while True:
        key = readchar().lower()
        # print(key)
        if key == "q":
            left_leg = left_leg+5
        elif key == "w":
```

(continues on next page)

(continued from previous page)

```
            left_leg = left_leg-5
        elif key == "z":
            left_foot = left_foot+5
        elif key == "x":
            left_foot = left_foot-5
        elif key == "i":
            right_leg = right_leg+5
        elif key == "o":
            right_leg = right_leg-5
        elif key == "n":
            right_foot = right_foot+5
        elif key == "m":
            right_foot = right_foot-5
        elif key == chr(32): # 32 for space
            print(right_leg,right_foot,left_leg,left_foot)
        elif key == chr(27): # 27 for ESC
            break

        right_leg_servo.angle(right_leg)
        right_foot_servo.angle(right_foot)
        left_leg_servo.angle(left_leg)
        left_foot_servo.angle(left_foot)
        # time.sleep(0.05)

    print("\nQuit")

if __name__ == "__main__":
    main()
```

4.10 Custom Action

In the previous project, we were able to give PiSloth custom steps, so how do we combine these steps into actions?

For example, have PiSloth make the step from the previous project and then return to the initial position.

Note: You can download and print the PDF Cartoon Mask for your PiSloth.

Step 1: Go to the `/home/pi/pisloth/examples` path.

```
cd /home/pi/pisloth/examples
```

Step 2: Open `custom_action.py` with the following command.

```
nano custom_action.py
```

Step 3: Modify the angle in `sloth.add_action()`, each group represents a step, and only 2 steps are set here. You can set multiple steps as needed.

```
sloth.add_action("my_action", [
    [ 0,-45  ,0, 40],
    [0,   0, 0,   0]
    ])
```

Step 4: Run this code.

```
sudo python3 custom_action.py
```

Code

Note: You can **Modify/Reset/Copy/Run/Stop** the code below. But before that, you need to go to source code path like `pisloth\examples`. After modifying the code, you can run it directly to see the effect.

```
from pisloth import Sloth
import time

sloth = Sloth([1,2,3,4])
sloth.add_action("my_action", [
    [ 0,-45  ,0, 40],
    [0,   0, 0,   0]
    ])

def main():
    sloth.do_action("my_action", 1, 80)
    time.sleep(1)

if __name__ == "__main__":
    while True:
        main()
```

CHAPTER

FIVE

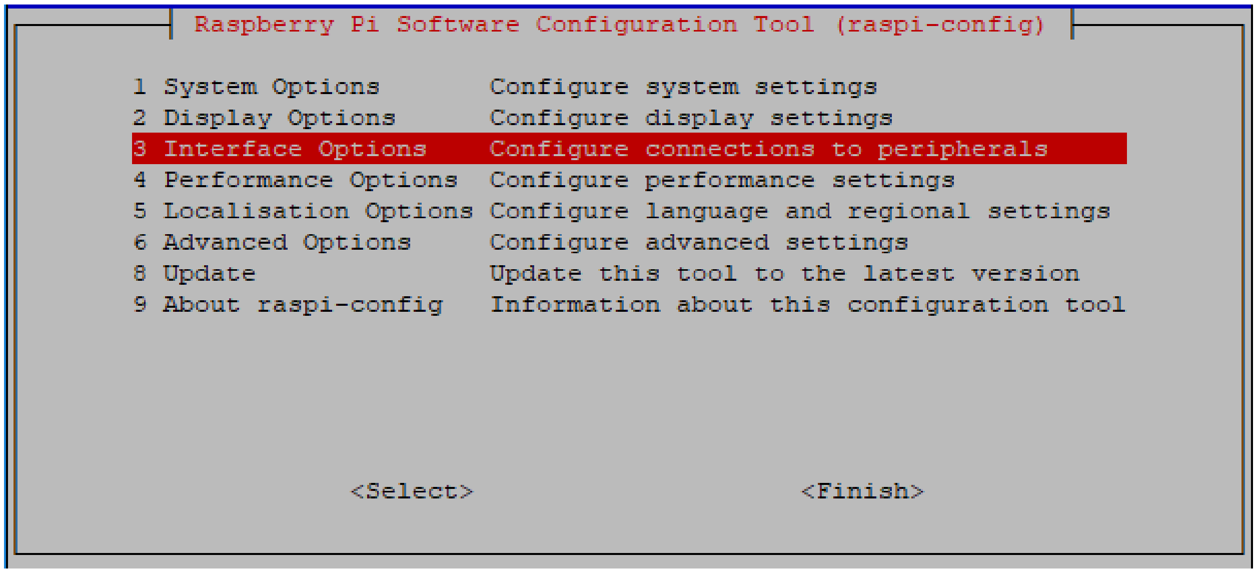
APPENDIX

5.1 I2C Configuration

Enable the I2C port of your Raspberry Pi (If you have enabled it, skip this; if you do not know whether you have done that or not, please continue).

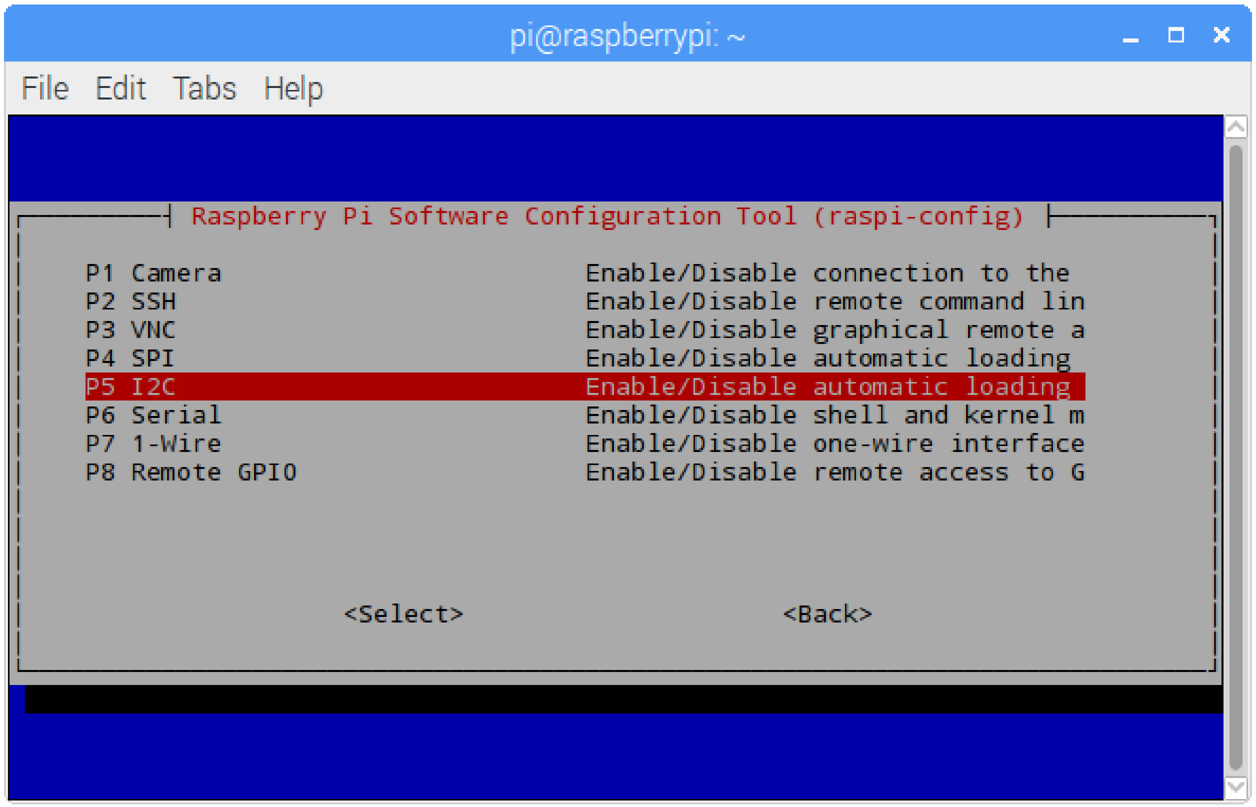
```
sudo raspi-config
```

3 Interfacing options



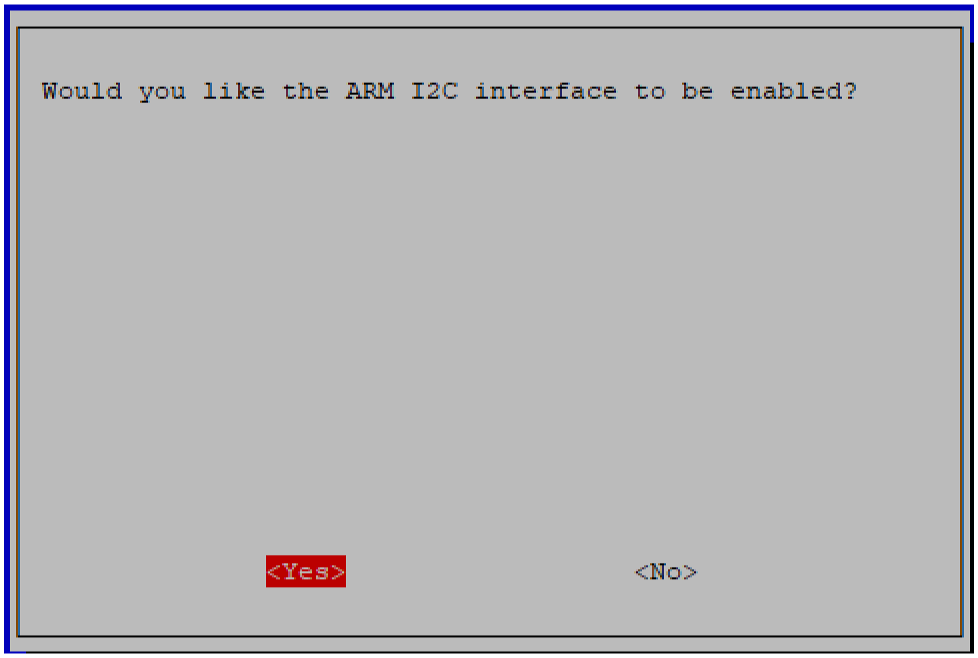
```
Raspberry Pi Software Configuration Tool (raspi-config)

1 System Options       Configure system settings
2 Display Options      Configure display settings
3 Interface Options    Configure connections to peripherals
4 Performance Options  Configure performance settings
5 Localisation Options Configure language and regional settings
6 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool

                <Select>                     <Finish>
```

P5 I2C



```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera            Enable/Disable connection to the
P2 SSH               Enable/Disable remote command lin
P3 VNC               Enable/Disable graphical remote a
P4 SPI               Enable/Disable automatic loading
P5 I2C               Enable/Disable automatic loading
P6 Serial            Enable/Disable shell and kernel m
P7 1-Wire            Enable/Disable one-wire interface
P8 Remote GPIO       Enable/Disable remote access to G

         <Select>                    <Back>
```

<Yes>, then <Ok> -> <Finish>.



```
Would you like the ARM I2C interface to be enabled?

            <Yes>                  <No>
```

5.2 Remote Desktop

There are two ways to control the desktop of the Raspberry Pi remotely:

VNC and **XRDP**, you can use any of them.

5.2.1 VNC

You can use the function of remote desktop through VNC.

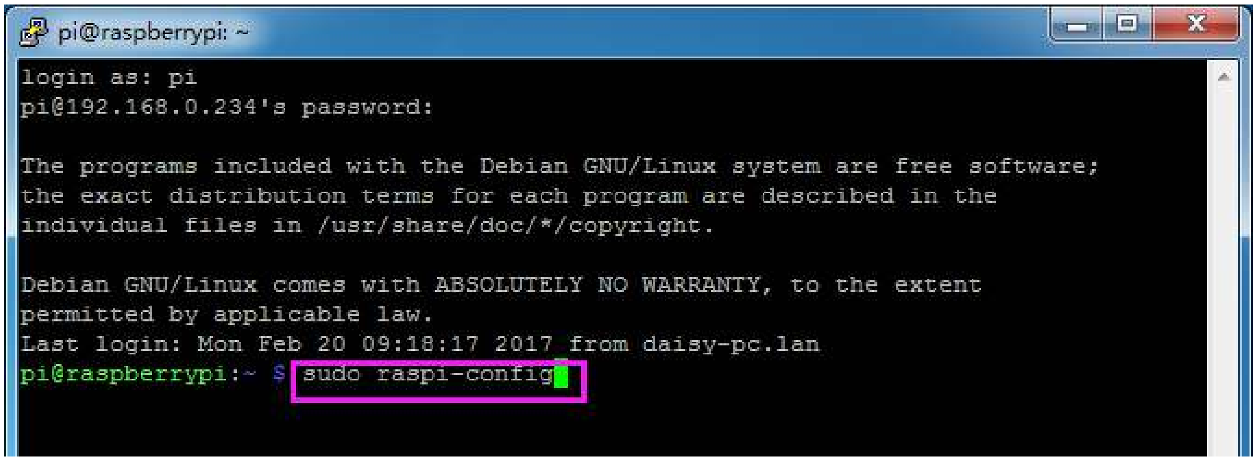
Enable VNC service

The VNC service has been installed in the system. By default, VNC is disabled. You need to enable it in config.

Step 1

Input the following command:

```
sudo raspi-config
```

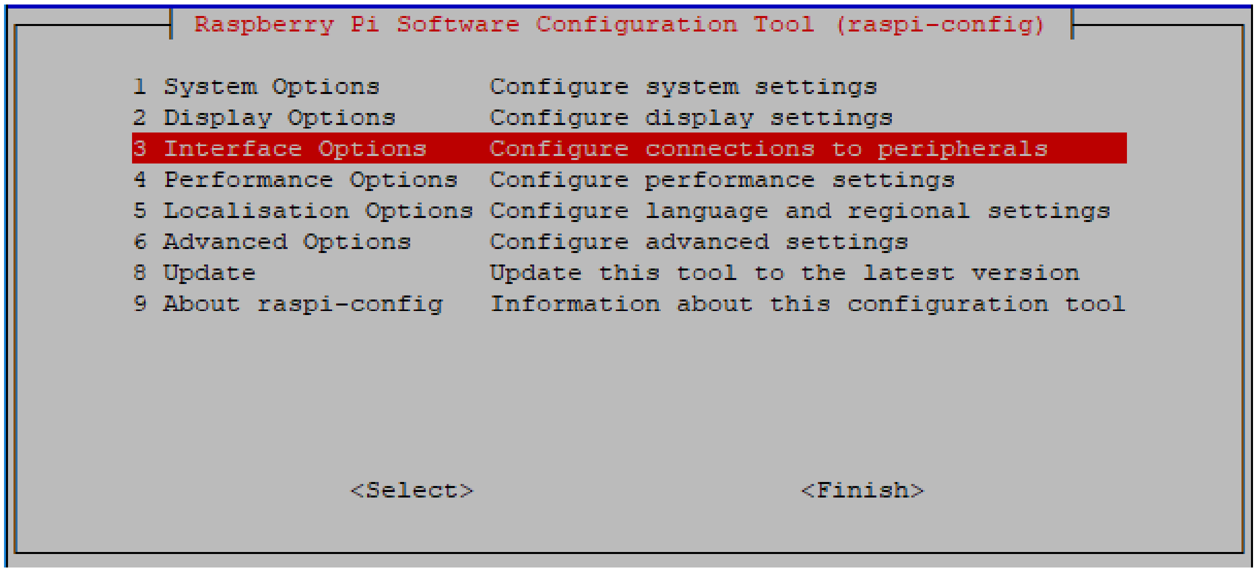


pi@raspberrypi: ~

```
login as: pi
pi@192.168.0.234's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Feb 20 09:18:17 2017 from daisy-pc.lan
pi@raspberrypi:~ $ sudo raspi-config
```

Step 2

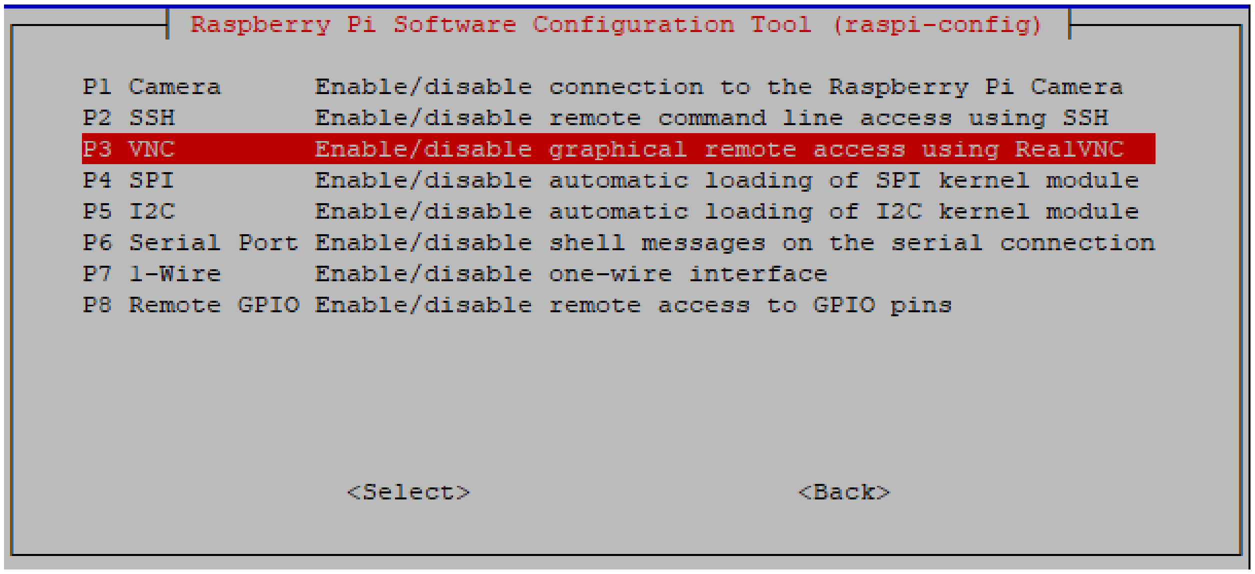
Choose **3 Interfacing Options** by press the down arrow key on your keyboard, then press the **Enter** key.



```
Raspberry Pi Software Configuration Tool (raspi-config)

1 System Options       Configure system settings
2 Display Options      Configure display settings
3 Interface Options    Configure connections to peripherals
4 Performance Options  Configure performance settings
5 Localisation Options Configure language and regional settings
6 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool

              <Select>                     <Finish>
```

Step 3

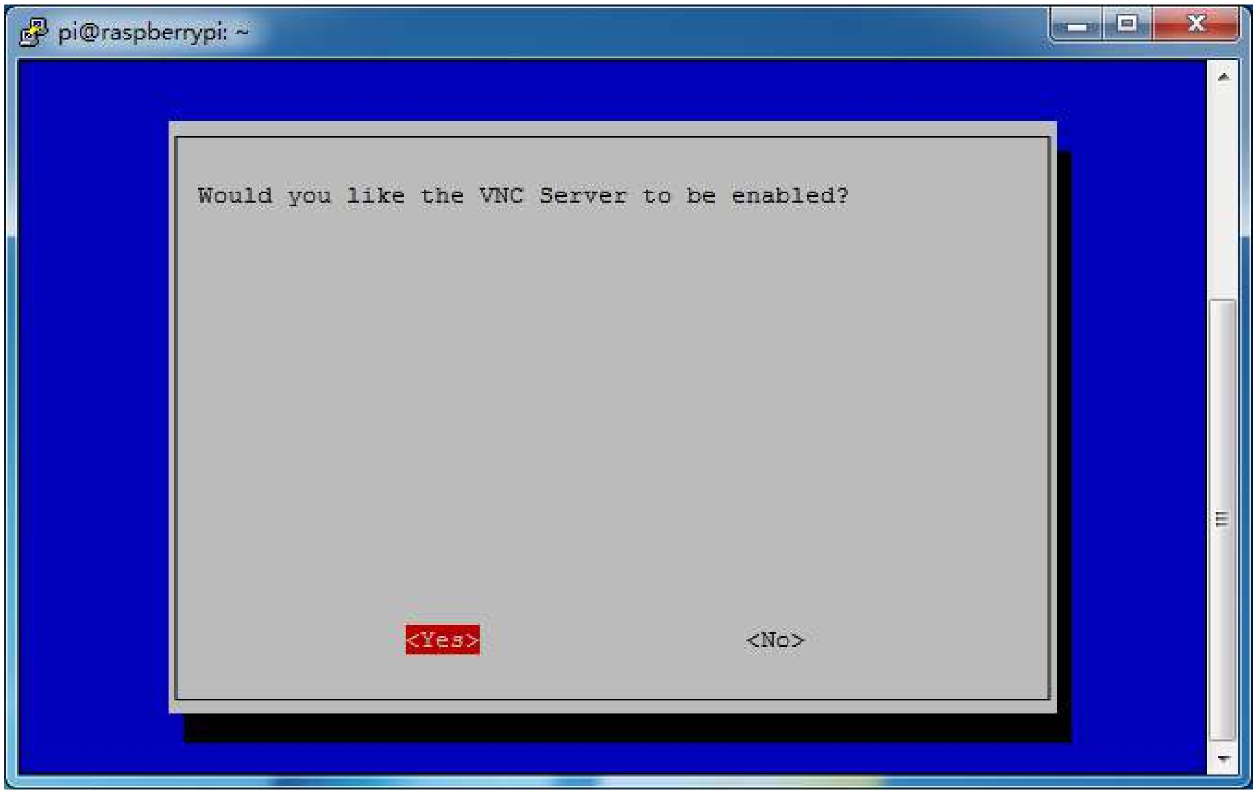
P3 VNC



```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera      Enable/disable connection to the Raspberry Pi Camera
P2 SSH         Enable/disable remote command line access using SSH
P3 VNC         Enable/disable graphical remote access using RealVNC
P4 SPI         Enable/disable automatic loading of SPI kernel module
P5 I2C         Enable/disable automatic loading of I2C kernel module
P6 Serial Port Enable/disable shell messages on the serial connection
P7 1-Wire      Enable/disable one-wire interface
P8 Remote GPIO Enable/disable remote access to GPIO pins

                  <Select>                     <Back>
```

Step 4

Select **Yes -> OK -> Finish** to exit the configuration.



```
pi@raspberrypi: ~

Would you like the VNC Server to be enabled?

             <Yes>                 <No>
```

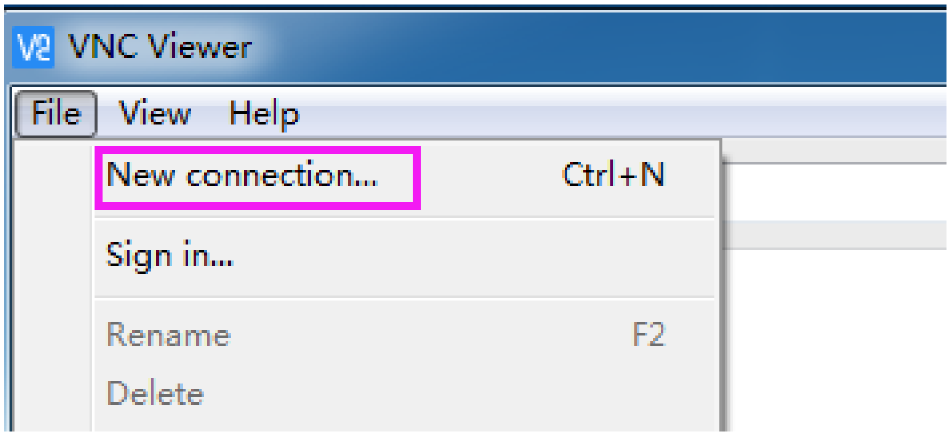
Login to VNC

Step 1

You need to download and install the VNC Viewer on personal computer. After the installation is done, open it.

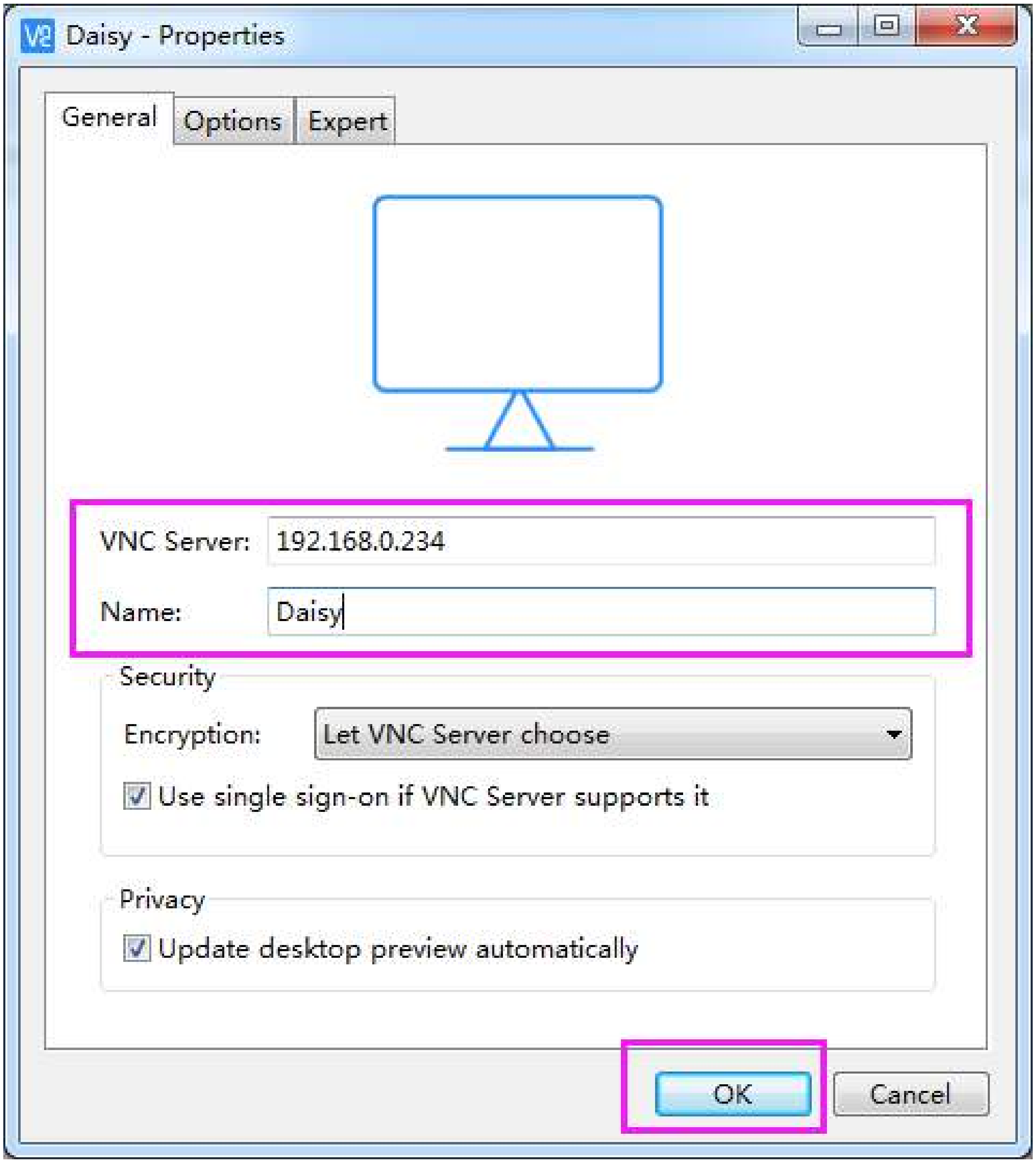
Step 2

Then select "**New connection**".



Step 3

Input IP address of Raspberry Pi and any **Name**.


Daisy - Properties

VNC Server: 192.168.0.234

Name: Daisy

Security

Encryption: Let VNC Server choose

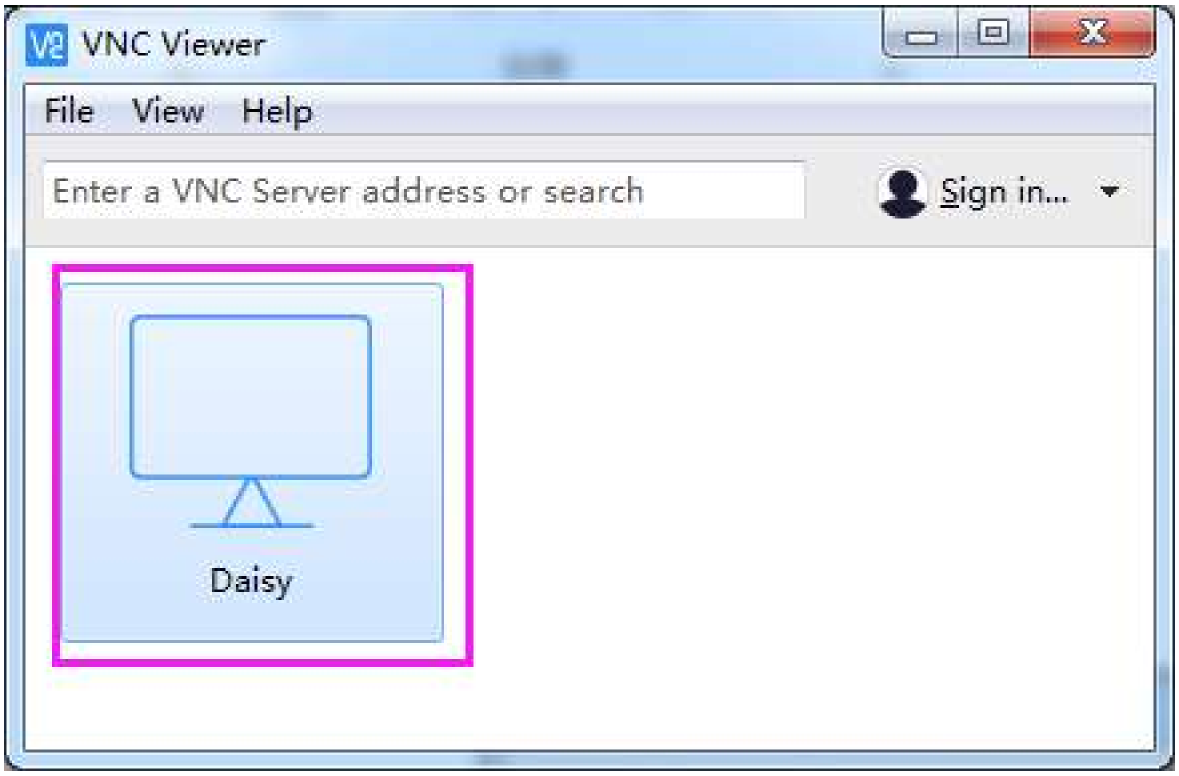
☑ Use single sign-on if VNC Server supports it

Privacy

☑ Update desktop preview automatically

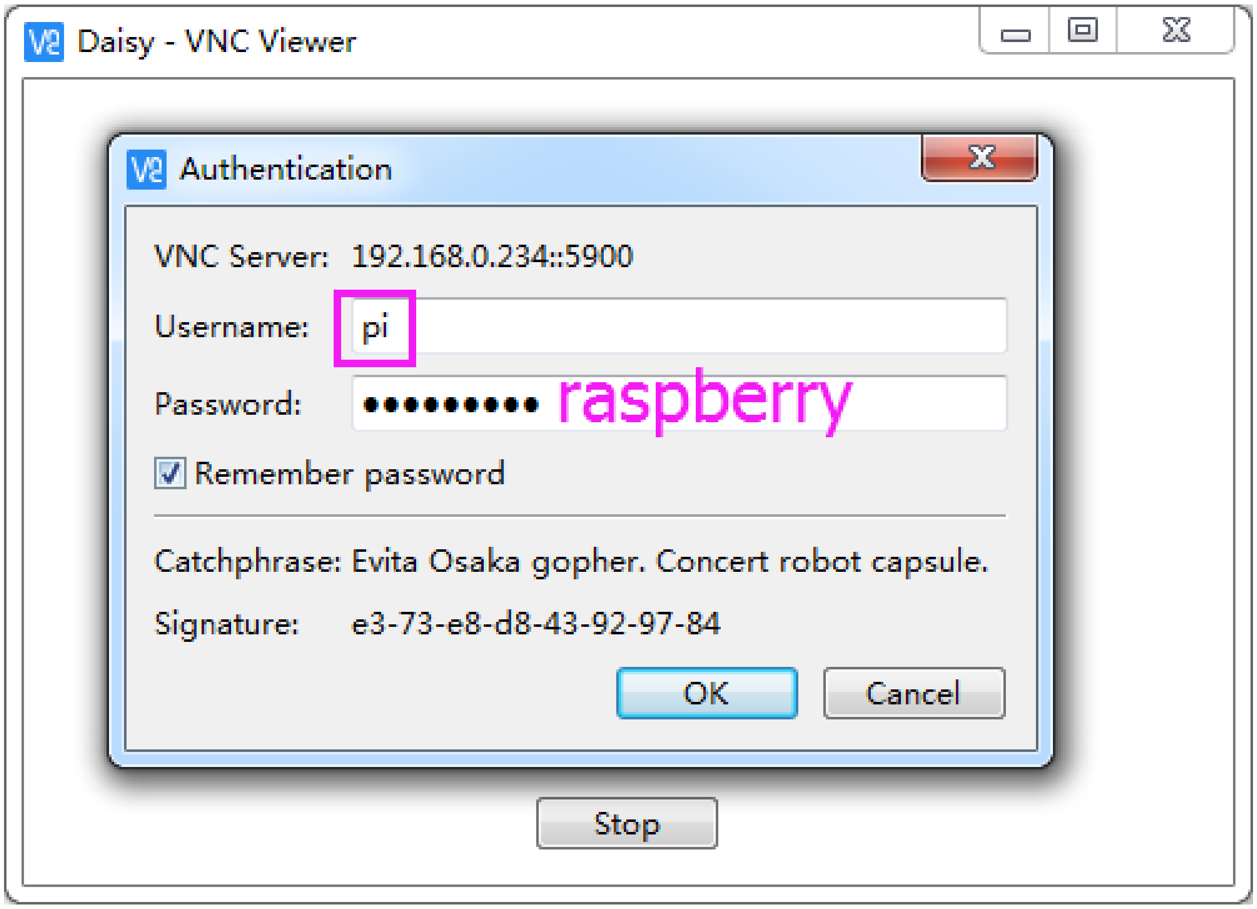
Step 4

Double click the **connection** just created:



Step 5

Enter Username (**pi**) and Password (**raspberry** by default).


Daisy - VNC Viewer

Authentication

VNC Server: 192.168.0.234::5900

Username: pi

Password: •••••••••• raspberry

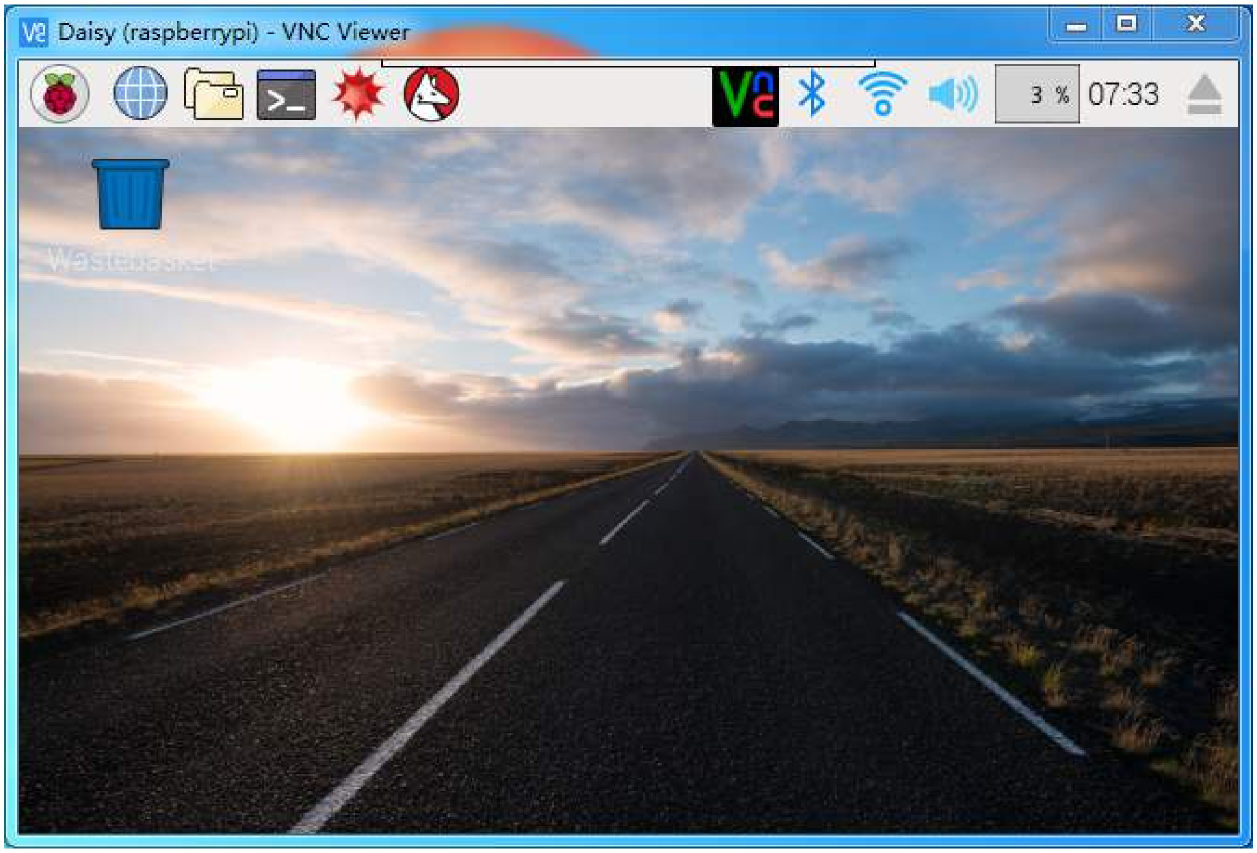
☑ Remember password

Catchphrase: Evita Osaka gopher. Concert robot capsule.

Signature: e3-73-e8-d8-43-92-97-84

Step 6

Now you can see the desktop of the Raspberry Pi:



That's the end of the VNC part.

5.2.2 XRDP

Another method of remote desktop is XRDP, it provides a graphical login to remote machines using RDP (Microsoft Remote Desktop Protocol).

Install XRDP

Step 1

Login to Raspberry Pi by using SSH.

Step 2

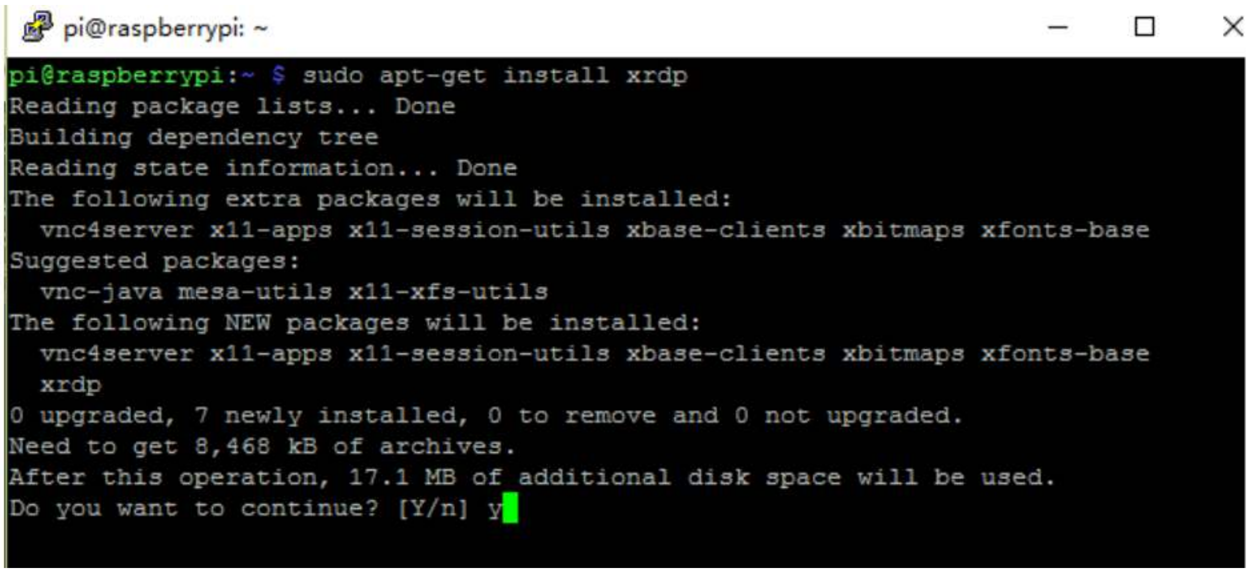
Input the following instructions to install XRDP.

```
sudo apt-get update
sudo apt-get install xrdp
```

Step 3

Later, the installation starts.

Enter "Y", press key "Enter" to confirm.



```
pi@raspberrypi:~ $ sudo apt-get install xrdp
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  vnc4server x11-apps x11-session-utils xbase-clients xbitmaps xfonts-base
Suggested packages:
  vnc-java mesa-utils x11-xfs-utils
The following NEW packages will be installed:
  vnc4server x11-apps x11-session-utils xbase-clients xbitmaps xfonts-base
  xrdp
0 upgraded, 7 newly installed, 0 to remove and 0 not upgraded.
Need to get 8,468 kB of archives.
After this operation, 17.1 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Step 4

Finished the installation, you should login to your Raspberry Pi by using Windows remote desktop applications.

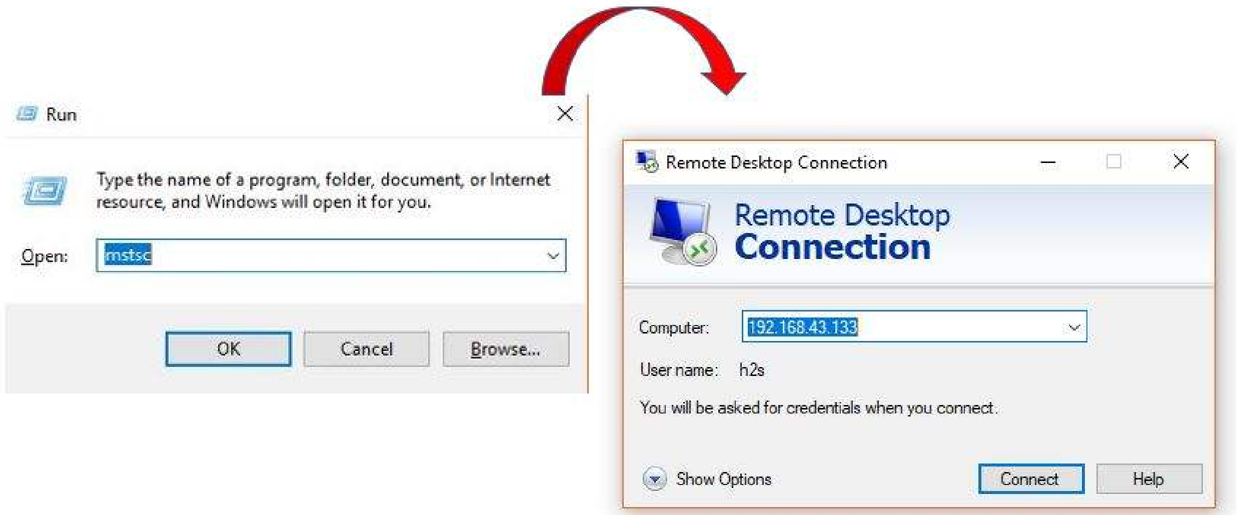
Login to XRDP

Step 1

If you are a Windows user, you can use the Remote Desktop feature that comes with Windows. If you are a Mac user, you can download and use Microsoft Remote Desktop from the APP Store, and there is not much difference between the two. The next example is Windows remote desktop.

Step 2

Type in "**mstsc**" in Run (WIN+R) to open the Remote Desktop Connection, and input the IP address of Raspberry Pi, then click on "Connect".



Run

Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: mstsc

Remote Desktop Connection

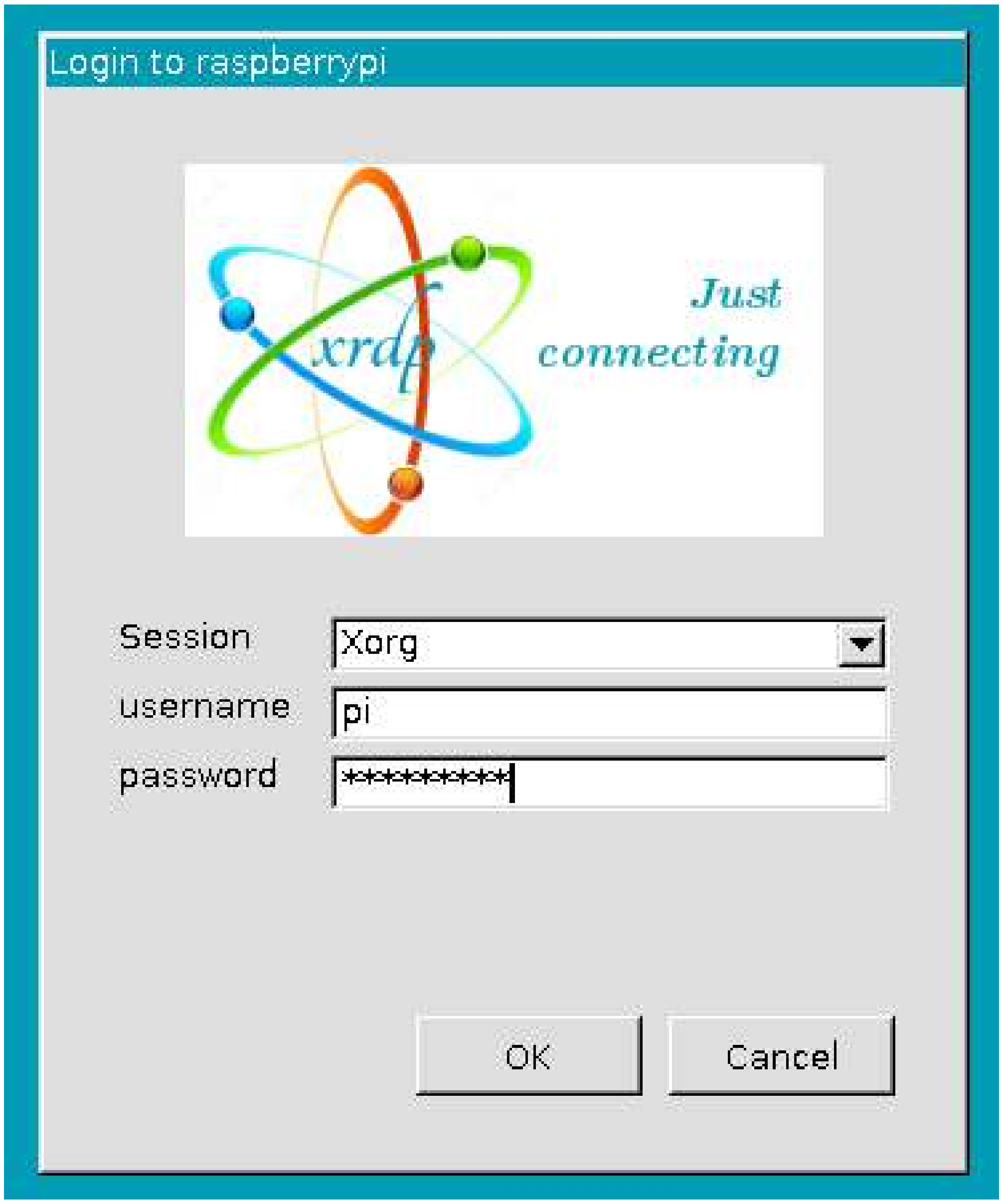
Computer: 192.168.43.133

User name: h2s

You will be asked for credentials when you connect.

Step 3

Then the xrdp login page pops out. Please type in your username and password. After that, please click “OK”. At the first time you log in, your username is “pi” and the password is “raspberry”.



Session Xorg
username pi
password **********

Step 4

Here, you successfully login to RPi by using the remote desktop.

5.3 About the Battery

Applicable Parameters

- 3.7V
- 18650
- Rechargeable
- Li-ion Battery
- Button Top
- No Protective Board

Note:

- Robot HAT cannot charge the battery, so you need to buy a battery charger.
- When the two power indicators on the Robot HAT are off, it means the power is too low and the batteries need to be charged.

Button Top vs Flat Top?

Please choose battery with button top to ensure a good connection between the battery and the battery holder.

Button Top	Flat Top

No protective board?

You are recommend to use 18650 batteries without a protective board. Otherwise, the robot may be cut power and stop running because of the overcurrent protection of the protective board.

Battery capacity?

In order to keep the robot working for a long time, use large-capacity batteries as much as possible. It is recommended to purchase batteries with a capacity of 3000mAh and above.

5.4 Filezilla Software

The File Transfer Protocol (FTP) is a standard communication protocol used for the transfer of computer files from a server to a client on a computer network.

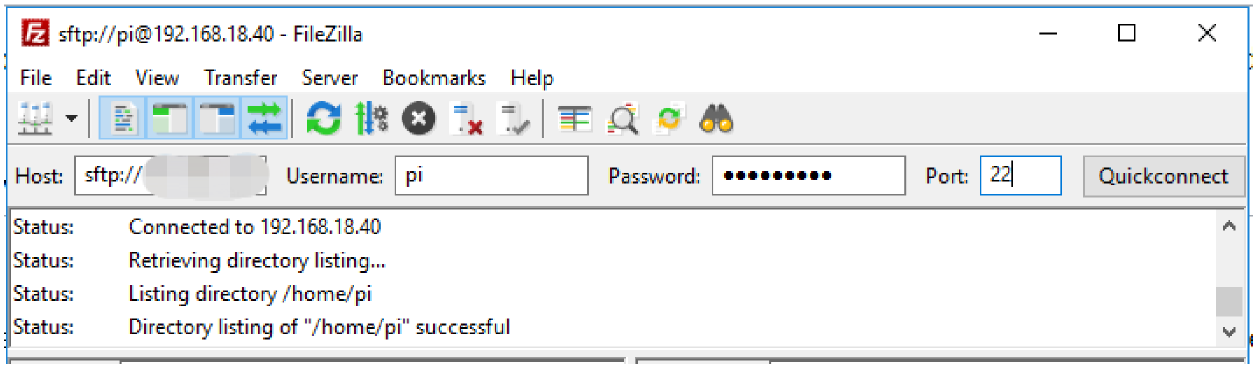
Filezilla is an open source software that not only supports FTP, but also FTP over TLS (FTPS) and SFTP. We can use Filezilla to upload local files (such as pictures and audio, etc.) to the Raspberry Pi, or download files from the Raspberry Pi to the local.

Step 1: Download Filezilla.

Download the client from Filezilla's official website, Filezilla has a very good tutorial, please refer to: Documentation - Filezilla.

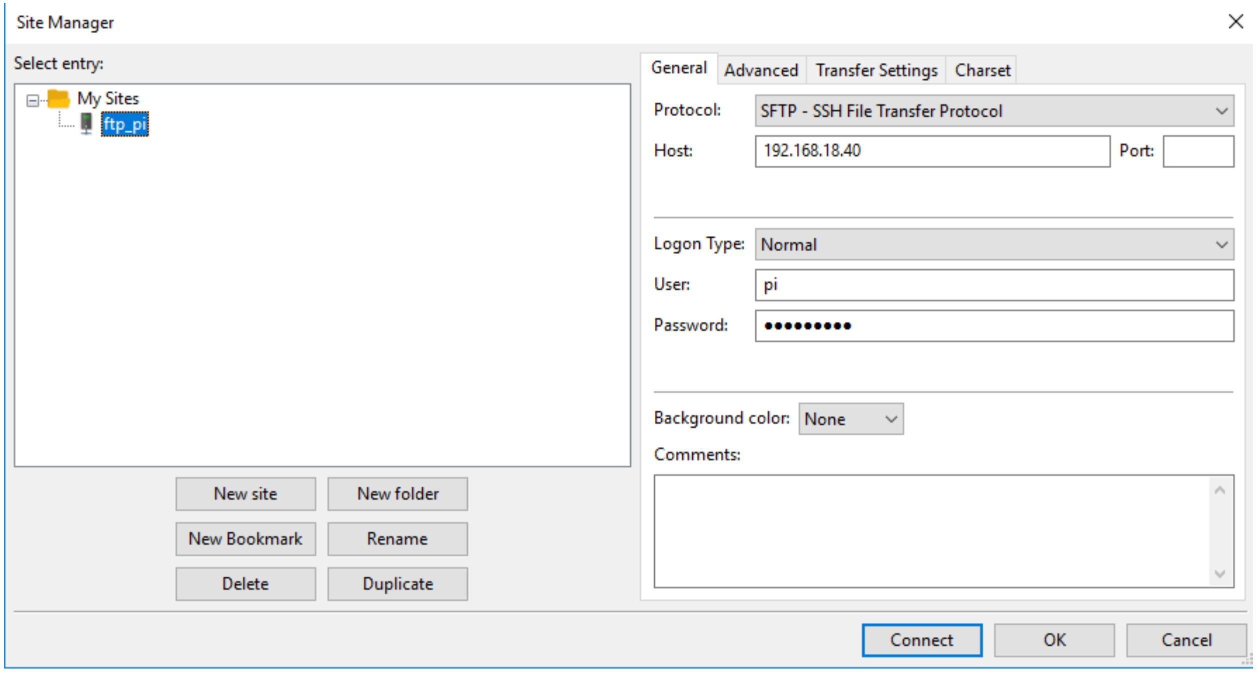
Step 2: Connect to Raspberry Pi

After a quick install open it up and now connect it to an FTP server. It has 3 ways to connect, here we use the **Quick Connect** bar. Enter the **hostname/IP**, **username**, **password** and **port (22)**, then click **Quick Connect** or press **Enter** to connect to the server.


sftp://pi@192.168.18.40 - FileZilla
File Edit View Transfer Server Bookmarks Help
Host: sftp://
Username: pi
Password: •••••••••
Port: 22
Quickconnect

```
Status:     Connected to 192.168.18.40
Status:     Retrieving directory listing...
Status:     Listing directory /home/pi
Status:     Directory listing of "/home/pi" successful
```

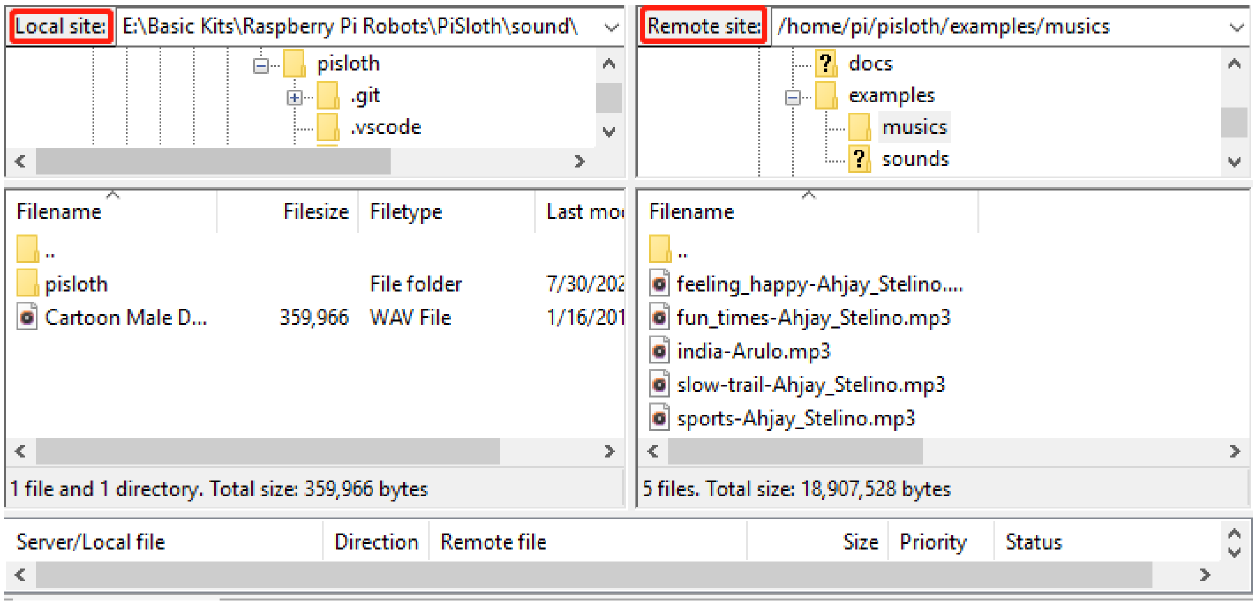
Note: Quick Connect is a good way to test your login information. If you want to create a permanent entry, you can select **File-> Copy Current Connection to Site Manager** after a successful Quick Connect, enter the name and click **OK**. Next time you will be able to connect by selecting the previously saved site inside **File -> Site Manager**.



Protocol: SFTP - SSH File Transfer Protocol
Host: 192.168.18.40 Port:
Logon Type: Normal
User: pi
Password: •••••••••
Background color: None

Step 3: Upload/download files.

You can upload local files to Raspberry Pi by dragging and dropping them, or download the files inside Raspberry Pi files locally.



Local site: E:\Basic Kits\Raspberry Pi Robots\PiSloth\sound\

Remote site: /home/pi/pisloth/examples/musics

Filename	Filesize	Filetype	Last mo
..			
pisloth		File folder	7/30/202
Cartoon Male D...	359,966	WAV File	1/16/201

1 file and 1 directory. Total size: 359,966 bytes

Filename
..
feeling_happy-Ahjay_Stelino....
fun_times-Ahjay_Stelino.mp3
india-Arulo.mp3
slow-trail-Ahjay_Stelino.mp3
sports-Ahjay_Stelino.mp3

5 files. Total size: 18,907,528 bytes

CHAPTER SIX

THANK YOU

Thanks to the evaluators who evaluated our products, the veterans who provided suggestions for the tutorial, and the users who have been following and supporting us. Your valuable suggestions to us are our motivation to provide better products!

Particular Thanks

- Len Davisson
- Kalen Daniel
- Juan Delacosta

Now, could you spare a little time to fill out this questionnaire?

Note: After submitting the questionnaire, please go back to the top to view the results.

CHAPTER

SEVEN

COPYRIGHT NOTICE