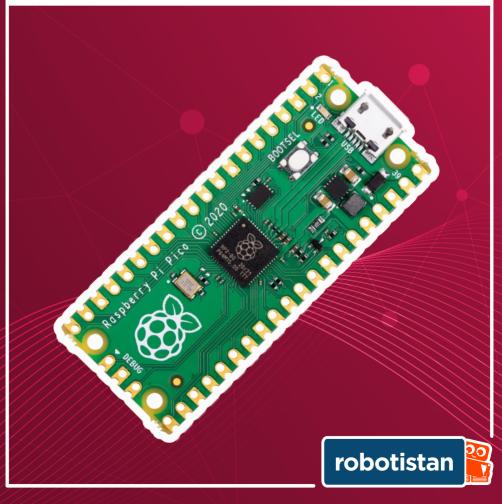


Raspberry Pi Pico Project Book



Welcome to the World of Electronics and Coding. Now that you have opened this book, you are eager to swin in the sea of wonder and learn new things.

Although learning new things in such matters is thought to be difficult, if you proceed step by step and with the right practices, you will realize that it is very simple. As long as the applications are made in the early stages, there will be places that do not sit down.You will overcome this problem as you practice.

It just takes a little patience, so you can learn robotic programming with an easy and correct roadmap, starting from the easy and moving towards more complex. If you want to watch more detailed video explanations of the applications, you can go to our Youtube channel by scanning the QR code at the back of the book. You can Access the codes writen in the booklet both from the description section of the related videos and from our blog page.

You can send us your set contents, applications, videos and any suggestions and questions you have in mind at "info@robotistan.com"

Robotistan Team

Contents

Introducing The Kit4
Introducing Raspberry Pi Pico6
Instaling Thonny IDE8
Installing Micropython Firmware on Raspberry Pi Pico11
Introducing Thonny IDE Interface14
Blink Internal LED16
LED Control with Button20
Temperature Measurement with Pico23
Weather Monitor26
Burglar Alarm
Using of Distance Sensor40

Introducing The Kit



Raspberry Pi Pico: Microcontroller we will perform our projects

Breadboard

It is our tool to test our circuits on the breadboard. It allows us to easily test the circuits we have built without soldering them to each other. It allows us to test the circuits we have designed be-fore transferring them to printed circuit boards or perforated pla-tes.

HC-SR501 PIR Sensor

PIR sensors are sensors used to detect the movement of living things in an environment. This tiny sensor is a module that can be used with many mic-rocontroller platforms such as Raspberry Pi, Arduino, ESP32, which you can use comfortably in various electronics, robotics and hobby applications.

This module, which has a digital output, gives a logic 0 output when it does not detect motion in the environment, and a logic 1 output when it detects motion.



Resistor:

In electrical circuits, resistance is the strain faced by an electric current flowing through a conductor.



2X16 LCD Screen:

It is a screen consisting of 2 lines and 16 columns that we can use in our pro-jects.



Jumper Cable:

The necessary cable to connect Raspberry Pi Pico and other sensors to the Breadboard.



LED:

The necessary cable to connect Raspberry Pi Pico and other sensors to the Breadbo-ard.



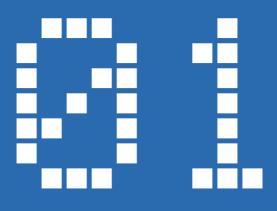
Buzzer:

Buzzer; It is a type of auditory warning device that works based on mechanical, elect-romechanical or piezoelectric principles. Buzzers, which have a lot of usage areas, ge-nerally work with the piezoelectric principle. Buzzers can be used in functions such as alarm, timer, confirmation and response warning depending on their usage areas.



HC-SR04 Ultrasonic Sensor:

Buzzer; It is a type of auditory warning device that works based on mechanical, electro-mechanical or piezoelectric principles. Buzzers, which have a lot of usage areas, generally work with the piezoelectric principle. Buzzers can be used in functions such as alarm, ti-mer, confirmation and response warning depending on their usage areas.



Introducing Raspberry Pi Pico







FORUM

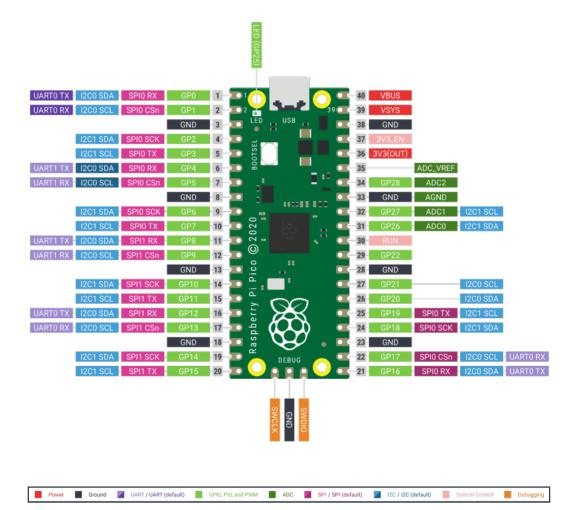




youtube.com/robotistan

Introducing Raspberry Pi Pico

Raspberry Pi Pico that was released by Raspberry Pi Foundation is a microcontroller that might use in our Embedded systems projects, prototyping or to the Micropython presented to us for both elect-ronic and in learning coding. Raspberry Pi Pico, which is quite powerful compared to Arduino Nano, stands out with its 32-bit Arm Cortex M0 + 133 MHz processor. Thanks to the temperature sensor on it, it will be very easy for us to make projects such as a weather monitor.



Installing Thonny IDE











youtube.com/robotistan

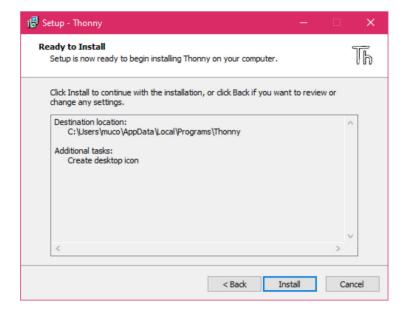
maker.robotistan.com

Firstly, You should download Thonny Ide which right version for your operating system on link https://thonny.org. In this blog, you will see the version for Windows.

D Theory Sylver El Note: × +			<i>с</i> − е х
			市 (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)
□ - -	Thonny Pythea IDE for beginners	Denolated service 3.3.6 for Vindens = bdis + Lipsy NRT On Winderse poer m-Lipsy a warrobig dialog from Defender. Aust click: "Anot of a dia" "Rate anywey".	
- 9	E Theory Fits Edit View Face Tools Hulp] 20 € 0 % 30 € 30 € 5	- u x	
0	femology s def Sectors Stars to Setters to Setters to	* None Volar *	
∴	relars factors i a testeril note s a lational l'anter a satural note praticita factorial al', bastalli fact		
e o	er factori If a s s name start FOR	And Deck(r): If it as its relates 1 class: Table: +====================================	
	Boll Control of Contro	Leaf saides	
	Features		
	Easy to get started. Thermy comes with Python 3.7 built in, so just one simple instiller is needed and you're ready to Lease programming. (You can also use a separate Python instillation, if necessary) The initial user interface is stripped of all features that more distant beainsers.	Choosy C X No Bar Total Nop L Bar D D D Makay	

You run the installation file you downloaded. Follow the these screenshots to complete the installa-tion process. You just need to click "Next".

🚏 Setup - Thonny — 🗆 🗙	15 Setup - Thonny — 🗆 🗙
Welcome to using Thonny!	License Agreement Please read the following important information before continuing,
<pre>1 impl t f ndm This wizard will retail Thomy 3.3.6 for your account. 2 guess = int(input 4 guess = int(input 5 while n != "guess 7 if guess < n: 8 print("gu 10 guess = i 10 elif guess > 11 print("gu 12 guess = i 13 else: 14 print("yc 15 Next> Cancel</pre>	Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation. The MIT License (MIT) Copyright (c) 2020 Aivar Anamaa Copyright (c) 2020 Aivar Anamaa Image: Copyright (c) 2020 Aivar Anamaa Bis offware and associated documentation files (the Software), to dash the Software without restriction, including without hemistation the right to use, copy, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: I cacept the agreement I do not accept the agreement I do not accept the agreement Image:
r BSetup - Thonny - 🗆 🗙	🖞 Setup - Thonny — 🗌 🗙
Select Destination Location Where should Thomy be installed?	Select Additional Tasks Which additional tasks should be performed?
Setup will install Thomy into the following folder. To continue, dick Next. If you would like to select a different folder, dick Browse. Exit Loss i purce Lips Costa i Local Programs (Thoma) Browse At least 98.7 MB of free disk space is required.	Select the additional tasks you would like Setup to perform while installing Thorny, then click Next.
< Back Next > Cancel	< Back Next > Cancel





Lastly, You click "Finish"

Installing Micropython Firmware on Raspberry Pi Pico







maker.robotistan.com

FORUM





youtube.com/robotistan

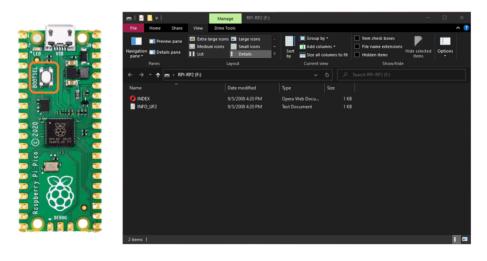
Raspberry Pi Pico is programmable in two programming language that MicroPython and "C/C++". You need to download the suitable "firmware" file for programming your Pico. 1- Just follow the instructions to download.

https://www.raspberrypi.com/documentation/mic-rocontrollers/micropython.html

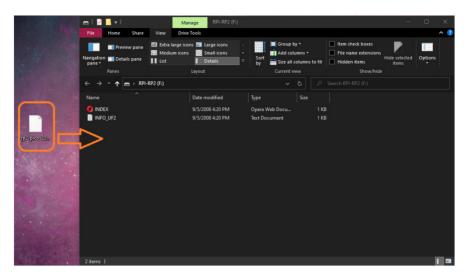
2- You click "Download UF2 file" and download the "firmware"

Image: Control of the control of th	iii Raspberry Pi Documentati: × +	Q _	
Forbid Replaymy PR No Replaymy PR No N	C 88 128 www.raspberrypi.com/	documentation/microcontrollers/micropython.html 😢 🕲 🦁 👂 ▷ 🗘 🔅	
Repose R	Microcontrollers		
Wind a Mod vy hodry Decomentation Decomentation The C/C++ SDK	Raspberry Pi Pico	You can program your Pico by connecting it to a computer via USB, then dragging and dropping a file onto it, so	
	Documentation Drag.and-Drop.MicroPython		
		 It will mount as a Mass Storage Device called RPI-RP2. Drag and drop the MicroPython UP2 file onto the RPI-RP2 volume. Your Pico will reboot. You are now running 	
 Drag and drop the MicroPython UF2 file onto the RPI-RP2 volume. Your Pico will reboot. You are now running 		MoroPython. 5. You can access the EPEL via USB Serial. Our MicroPython documentation contains step-by-step instructions for connecting by your Proc and programming it in MicroPython.	
 It will recent as a Mass Storage Device called RPI-RP2. Ing and drop the MicroPython UF2 (Me onto the RPI-RP2 volume. Your Pico will reboot. You are now running MicroPython. You can access the REPL via USB Serial. Our MicroPython documentation contains step-by-step instructions 		NOTE	
 It will recurt as a Mass Storage Device called RPI-RP2. Drag and drop the MicroPython UF2 file onto the RPI-RP2 volume. Your Pico will reboot. You are now running MicroPython. You can access the EPD, via USB Binal. Our MicroPython documentation contains etsp-by-step instructions for connecting to your Pico and programming it in MicroPython. 		If you're new to MicroPython, our official quide, "Get started with MicroPython on Raspberry Pi Pico", is a great	

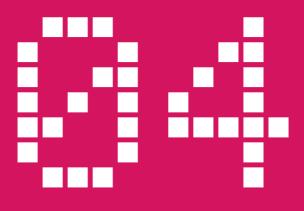
3- Connect to our computer by holding down the "BOOTSEL" button on your Pico. You will see the folder when you connected.



4- Copy the "*.UF2" file that downloaded on that folder.



When the copying process is finished, the folder will close automatically.



Introducing Thonny IDE Interface







FORUM





youtube.com/robotistan

Introducing Thonny IDE Interface

Unlike Arduino, Raspberry Pi Pico has its own memory ,and library or scripts what you want can be uploaded into the this memory with Thonny IDE. There are some useful things in Thonny IDE When you open the Thonny IDE, an interface like in the photo will welcome you. The functions of the tabs at the top, the section I have included in the orange box, respectively;

- Files: On this tab, you can save scripts what you wrote, open the scripts you have written or new Project file

- Edit: on this tab you can undo, forward, copy or select all in scripts what you have written

- View: On this tab ,you can customize the interface

- Run: on this tab, you can run code you have written, choose Interpreter for your Rasp-berry Pi Pico or operate like Debug on code

- Tools: On this tab, you can manage packets or edit Thonny settings

- Help: On this tab, you can get information about Thonny version or view help content regarding your issue.

1. You can open a new file with this icon

- 2. You can open scripts you have saved with this icon
- 3. You can save the script with this icon
- 4. You can run the script with this icon abi bu ne salak mı adamlar?
- 5. You can stop the script with this icon

Ite Thonny - <untitled> @ 1:1 File Edit View Run Tools Help</untitled>	-		×
1			
Shell 🕱			
>>> 			
MicroPy	thon (Ras	pberry Pi	Pico)

Blink Internal LED







maker.robotistan.com



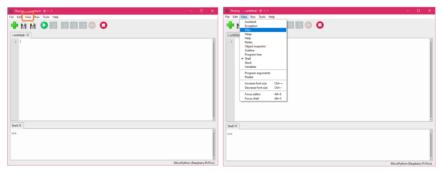


youtube.com/robotistan

You adjust the Interpreter Settings on Thonny IDE for using your Raspberry Pi Pico. Click the the run tab at the top, then click Select Interpreter. Select "MicroPython (Raspberry Pi Pico)" as the "Interpreter" and "Try to detect port automatically" as the port from the window that opens.

🖁 Thonny - <un< th=""><th></th><th></th><th></th><th></th><th>- 0</th><th></th><th>🚡 Thonny</th><th>options</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></un<>					- 0		🚡 Thonny	options									
	Run Tools Help					- 1			1								
<untitled> * X 1</untitled>	Select interpreter Run current script Debug current script (ricer) Debug current script (faster) Debug current script (birdseye)	F5 Ctrl+F5 Shift+F5 Ctrl+Shift+8				3	Which i	Interpreter interpreter o Python (Rasp	r device :	hould Tho		Run & Debu for running	-		Shell	Assistan	t
	Step over Step into Step out Resume Run to cursor Step back	F6 F7 F8 Ctrl+F8 Ctrl+B					Details Conr (look	ect your de	vice to th	e compute e, "USB Seri	ial" or "I	lect correspo JART"). oper USB dri	-		ow		
	Run current script in terminal Dock user windows Pygame Zero mode	Ctrl+T					Port	y to detect p			natur pr	oper oso un					
	Stop/Restart backend Interrupt execution Send EOF / Soft reboot Disconnect	Ctrl+F2 Ctrl+C Ctrl+D															
Shell X						1											
														Ins	tall or i	update fi	rmv
				MicroDuti	hon (Rasaherry Pi Pi											ОК	

Add "Files" tab that will contribute on the interface.Click the the "View" tab at the top and select "Files".



After that, you will see a tab like these screenshots..

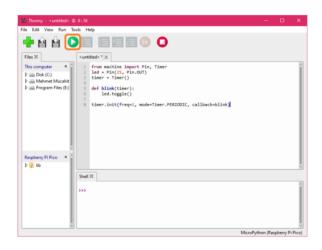
Thonny - <untitled> 0 File Edit View Run To</untitled>		-		×
🕂 🖞 🖞 🕻				
Files X This computer This computer This Computer This (C) D and Michaelt Machelt D and Program Files (E)	1			Î
Respberry Pi Pico = C D 😺 lib	9et X			
	>>> 	Python (Ras	pberry P	i Pico)

Blink Internal LED

Type the following code into Thonny IDE this code provides the Internal LED to blink every half second When you examine this code, you wrote the code "from machine import Pin, Timer" so that be able to access the hardware on the Pico. In line 2, you stated that the internal LED on the Pico is connected to pin 25. In line 3, you started timer .In line 5, you defined a blink function. This is a simple function that makes the LED light. In line 8, you ran the timer. We specified the frequency of the pro-cess with "freq" in parentheses, the mode of the timer with the "mode" and the function we would call with the "callback".

```
from machine import Pin, Timer
led = Pin(25, Pin.OUT)
timer = Timer()
def blink(timer):
    led.toggle()
timer.init(freq=2, mode=Timer.PERIODIC, callback=blink)
```

Upload this code to your Pico. Let's click on the green "Run" icon at the top.



It asks where you want to save the code. If save the code to the computer and run it, the code will only run when the Pico is plugged into the computer. For the code to always run. You need to click on the "Raspberry Pi Pico" option and save it to your Pico.

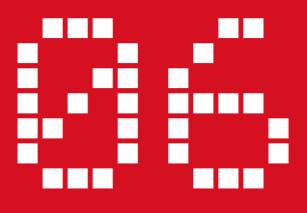
the Thonny - <untitled> ☺ File Edit View Run Too</untitled>		-		×
Files X This computer = D = Disk (C:) D = Mehmet Mücahit	<pre>cutted>*X cutted>*X i from machine import Pin, Timer i data Pin(CS; Pin, Ort) i timer = Timer() </pre>			
D in Program Files (E)	<pre>5 def blink(timer): 1ed.toggl() 7 timer.init(fre 1. Where to save to? 7 This computer 7 This computer</pre>			
Raspberry Pi Pico =	Raspberry Pi Pico			
	MicroPy	thon (Ra:	pberry P	i Pice

After clicking on the "Raspberry Pi Pico" text, a window like in the photo will open. In this window, it asks us to which location you want to save the code you wrote in Pico. You can save it directly in Pico. For this, you need to write a name and file extension to save the code youhave written. Since you are working with MicroPython, your file extensions will always be "* .py". I typed "blink.py" as the filename. You can also type any file name you want. After typing the file name, you can save and run the code by clicking the "OK" button.

The Thonny - <untitled></untitled>		-		\times
File Edit View Run	Tools Help			
🕂 🗎 🖄 🛉	🗽 Save to Raspberry Pi Pico	×		
Files X	us save to haspberry Privico	<u> </u>		
	Raspberry Pi Pico			
This computer	Name Size (bytes)			- 1
 Disk (C:) Mehmet Mücah 	J lib			- 1
Program Files (E)	•			- 1
				- 1
				- 1
				- 1
				- 1
				- 1
				- 1
	· · · · · · · · · · · · · · · · · · ·	5		- 1
	File name: blink.py OK Cancel			- 1
Raspberry Pi Pico	ok carce			- 1
Þ 🔑 lib 🕓				-
	Shell X			
				-
	>>>			- 1
				- 1
				- 1
				- 1
	MicroP	ython (Ra	spberry P	i Pico)

When the code runs, the internal LED on your Pico will blink and flash at half-second intervals.





LED Control with Button







FORUM





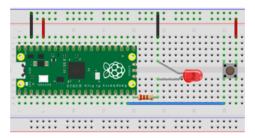
youtube.com/robotistan

Once you lit LEDs, but this time, you need to realize this project in order to warm up our Pico's hardware and breadboard usage.

Necessary Materials:

- Raspberry Pi Pico
- Breadboard
- Push Button
- LED
- Male Male Jumper
- 220 Ω or 330 Ω Resistor

When you look at the pin diagram of your Pico, the 38th pin is "GND", ie the ground pin, the 36th pin is the "3V3 (OUT)" pin. You will use these two pins frequently in your projects. First, build the circuit on the breadboard according to the diagram below.



Then open your Thonny IDE and edit the "blink" code you have written in the previous project. Open it by double clicking on the "blink.py" file on the left side of Thonny IDE. The codes you have written in the previous project will be opened. Now you will delete these codes completely and write new code.

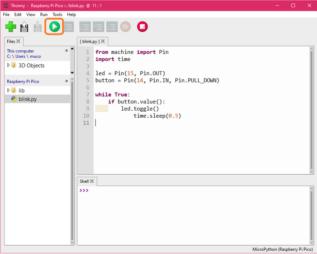
1. Thonny - Raspberry Pi Pico :: /blink.py @ 10 : 1	-		×
File Edit View Run Tools Help			
Files X (blink.py)X			
This computer C:\Users\muco 1 from machine import Pin, Timer 2 led = Pin(25, Pin.OUT)			Î
▷ JD Objects 3 timer = Timer()			
Raspberry Pi Pico = 5 def blink(timer):			
<pre></pre>	nk)		(
Shell 30			
			Î
Mic	oPython (R	aspberry Pi	Pico)

LED Control with Button

When you examine the codes below, you defined the hardware of Pi-co with the code "from mac-hine import Pin" as in the previous project and added the "time" library. With the code "led = Pin (15, Pin.OUT)", you assigned the 15 nu-mara pin as the pin you connected to the LED. With the code "but-ton = Pin (14, Pin.IN, Pin.PULL_DOWN)", you defined the pin number 14 to which you connect the button as the input pin. Then you created a "while loop". In this cycle, it will change the status of the led according to the value read from the button. For example, if press the button once while the LED is off, it will turn on, if the LED is on, it will turn off when you press the button. When press and hold the button with the code "time.sleep (0.5)", the LED will flash and flash at half-second intervals.

```
1 from machine import Pin
2 import time
3
4 led = Pin(15, Pin.OUT)
5 button = Pin(14, Pin.IN, Pin.PULL_DOWN)
6
7 while True:
8 if button.value():
9 led.toggle()
10 time.sleep(0.5)
```

After writing the codes, press the "Run" icon at the top and save your codes to your Pico and run them.



Now you will be able to turn the LED on and off by pressing the button. Move on to the next project.

Temperature Measurement with Pico







FORUM





youtube.com/robotistan

As I mentioned at the beginning, Raspberry Pi Pico has an internal temperature sensor. So why not take a temperature measurement?

First, open the Thonny IDE and open a new file and write the following codes. Now, when you examine the codes, you have added the necessary "machine" and "utime" libraries as usual. Since made analog readings with the temperature sensor, you defined that it was connected to pin number 4 with the code "sensor_temp = machine. ADC (4)", then you converted the 16-bit data from the sen-sor into voltage data that could be meaningful for you with the code "conversion_factor". Since the Pico pin gives 3.3 volts, you divided 3.3 volts by 216 - 1 = 65535. Now, when you examine the While loop, read the data from the sensor. In the code on line 8, multiply the data from the temperature sensor with the "conversion_factor" you just wrote, and write the temperature data in terms of vol-tage. You convert the data you have obtained to "Celsius" in the 9th line. Finally, print the temperature data to Shell with the code "print (temp)".

```
1 import machine
2 import utime
3
4 sensor_temp = machine.ADC(4)
5 conversion_factor = 3.3 / (65535)
6
7 while True:
8    reading = sensor_temp.read_ul6() * conversion_factor
9    temp = 27 - (reading - 0.706)/0.001721
10    print(temp)
11    utime.sleep(2)
```

Now save these codes to your Pico and run them, then press the "Run" icon at the top.



It asks where you want to save the code. For the code to always run, you need to click on the "Raspberry Pi Pico" option and save it to our Pico.

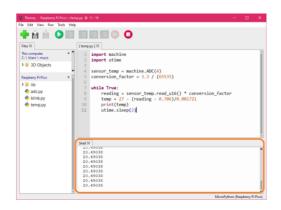
Temperature Measurement with Pico

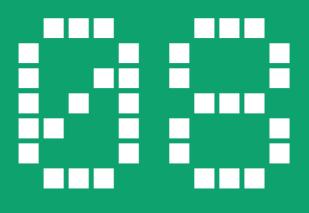
🖶 🛍 🛍 โ Ree X	contract * X		
Norman Annual Stream Ann	1 import machine 2 import utime 3 sensor_temp + machine.AOC(4) 5 conversion_factor = 3.3 / (05535) 6 while True: 8 read (b Whetewarts) 9 temp 9 temp 9 temp 9 temp 10 print 11 time 11 time 11 time 12 time 13 time 14 conversion_factor 14 conversion_factor 15 conversion_factor 16 conversion_factor 10 print 11 time 11 time 12 time 13 time 14 conversion_factor 14 conversion_factor 14 conversion_factor 15 conversion_factor 16 conversion_factor 17 time 18 conversion_factor 19 conversion_factor 10 print 11 time 11 time	or.	

Write "temp.py" as the file name, click the "OK" button and now you can read the temperature data from Shell.

The Thonny - <untitled> @ 11</untitled>				×
File Edit View Run Tools				
🕂 台 台 🔘	🔣 Save to Raspberry Pi Pico	×		
Files X	Raspberry Pi Pico	9		
This computer C: \ Users \ muco	Name Size (bytes)			Î
3D Objects	 Ø lib ● adcpy 636 			
Raspberry Pi Pico	📀 blink.py 192			- 11
▶ 🖟 lib 🔮 adc.py � blink.py	Fit name frem pol	actor		
	20:0224 20:0224 20:0224 20:0224 20:0224 20:0224 20:0224 20:0224 10:021	th RP2040	2	

In the Shell window at the bottom, you can see the temperature data from your Pico every 2 seconds.





Weather Monitor







Forum





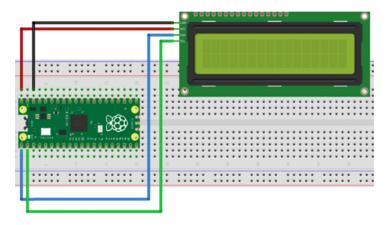
youtube.com/robotistan

In this project, you will make a weather monitor using a 2X16 LCD screen. In this project, you will learn how to add a library and automatically save the temperature data you measure in a text file.

Necessary materials:

- Raspberry Pi Pico
- Breadboard
- 2X16 LCD Screen
- Male Female Jumper

First, build your own circuit on the breadboard according to the circuit below.



Now download the necessary library to use our LCD screen from https://github.com/T-622/RPI-PICO-I2C-LCD. When open the link, click the green button that says "Code" and then click the "Download ZIP" button to download the library.

0	O GitHub - T-622/RPI-PICO-II × +	۹ _ D X
	< > C 🔀 📼 🗎 github.com/T-622/RPI-PICO-12C-LCD	@ ♥ ▷ ♡ ୭ ଡ ± ቹ
<u>ි</u> ය	Why GitHub? Team Enterprise Explore Marketplace Pricing Search	🖉 Sign in Sign up
-	G T-622 / RPI-PICO-I2C-LCD	is 🟠 Star 14 💱 Fork 5
9 _	↔ Code ① Issues 11 Pull requests ⓒ Actions Projects ③ Security 🗠 Insights	
	P main → P 2 branches O tags Go to file	About
	T-622 Update README.md Clone O HTTPS_GH-lub CU	This is a project which adapts code from another user to allow usage of the PCF8574 I2C lcd
	I2C Icd_bb.jpg Add files via upload https://github.com/T-622/RPI-PICO-I2C-LCD. II	backpack for either 20x4 or 16x2
	README.md Update README.md Use Git or checkout with SVN using the web URL.	Co screens.
	Icd_api.py Add files via upload Gpen with GitHub Desktop	L Keadme
\$	D pico_j2c_lcd.py Add files via upload	Releases
	pico_i2c_lcd_test.py Better date / time for Download ZIP	No releases published
	README.md	Lan constant frances and
	RPI-PICO-I2C-LCD	Packages No packages published
	This is a second at which extends and a form another contact allow on an afairs. BOENETA IDO had	×

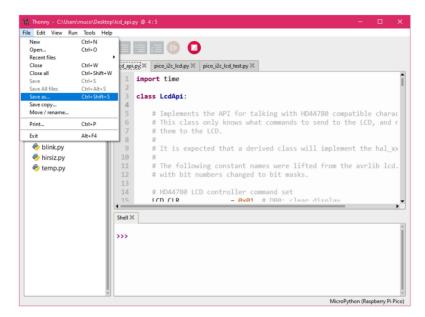
Open the downloaded "* .zip" file and extract the "lcd_api.py", "pico_i2c_lcd.py", "pico_i2c_lcd_test.py" files to the desktop.

			RPI-I	PICO-I2C-LCE)-main.zip									-	×
			Dosya	Komutlar A	raçlar Sık K	ullanılanlar	Seçene	kler Yard	lim						
			Ekle	Dizine Çık	art Test Et	Göster	Sil		Sihirbaz	Bilgi	VirüsTara	Açıklama	1 SFX		
TI	ĥ			RPI-PICC											 ~
lied o			İsim			^			3b03efa5	idea5e27	11bb927	71ab3b20e	845d6f	9a5	^
	1														
	-		The local	pi.py i2c_lcd.py											
TA				i2c_lcd_test.p	y J										
piece iž	Re Lee		READ												
			E Scher	natic.jpg											
Te															
Jü	'n														
(2000-02	<u>e llem</u>	1													
and the															
and the															
and the second															
1.1.1.1.1.															
and the second															
			<					>	<						>
			C						1	Ionlam 358	183 havt 5	dosva			

Open them in Thonny IDE by double clicking on these files that you extract to the desktop.

Thonny - C:\Users\muco\Desktop	\lcd_api.p	y @ 187:1 ── ×
File Edit View Run Tools Help		
🖶 🖄 📩 🕒 🔲	JE	
Files 🗶	Icd_api	py 🗶 pico_i2c_lcd.py 🗶 pico_i2c_lcd_test.py 🗶
This computer = C: \ Users \ muco	1	import time
Image: Provide the second s	3	class LcdApi:
Raspberry Pi Pico =	4	# Implements the API for talking with HD44780 compatible charac
2 lib	6	# This class only knows what commands to send to the LCD, and r
🗢 adc.py	7	# them to the LCD.
	8	#
🤣 blink.py	9	<pre># It is expected that a derived class will implement the hal_x></pre>
🔶 hirsiz.py	10	#
🗢 temp.py	11 12	<pre># The following constant names were lifted from the avrlib lcd. # with bit numbers changed to bit masks.</pre>
	13	# with bit numbers changed to bit masks.
	14	# HD44780 LCD controller command set
	15	ICD CLR - AvA1 # DRA: clear display
	-	•
	Shell 🛛	
	þ>>	
		MicroPython (Raspberry Pi Pico)

Now save these files to the "lib" folder in your Pico in order, click on the "Files" tab on the top and then click on "Save as ...".



It asks where you want to save the library file. You will choose the Raspberry Pi Pico.

Thonny - C:\Users\muco\Desktop	\lcd_api.py @ 4:5 —		×
File Edit View Run Tools Help			
Files 🗶	lcd_api.py X pico_i2c_lcd.py X pico_i2c_lcd_test.py X		
This computer C: \ Users \ muco ▷ ♫ 3D Objects	1 import time 2 3 class LcdApi: 4		Î
Raspbery Pi Pico = lib e adc.py blink.py blink.py temp.py temp.py	<pre>5 # Implements the API for talking with HD44780 compatib 6 # This class only knows what commands to send to the L 7 # them to the L(D, 8 # 10 # It 11 # The 12 # wit 13 # HD4 15 ICO ShelX</pre>	CD, and	r xx
v	MicroPython (Raspberry Pi	Pico)

Do the same for the other two files.

Thonny - C:\Users\muco\Deskt	op\pico_i2c_lcd.py @ 2:1	– 🗆 🗙
File Edit View Run Tools He	lp	
🕂 🛍 📩 🕒 🗖		
Files 🗶	[lcd_api.py] X pico_i2c_lcd.py X pico_i2c_lcd_test.py X	
This computer ≡ C: \ Users \ muco	🚡 Save to Raspberry Pi Pico 🛛 🗙	î
D 3D Objects	Parnhamy Di Diro	
	Raspberry Pi Pico = 1	
Raspberry Pi Pico ■ マ 💫 lib	Name Size (bytes)	
	🗢 lcd_api.py 6996	
e adc.py		
🌏 blink.py		
hirsiz.py		
🎨 temp.py		
	File name: pico i2c lcd.pvl OK Cancel	
	File name: pico_i2c_lcd.py OK Cancel	
	MicroP	ython (Raspberry Pi Pico)
🌃 Thonny - Raspberry Pi Pico :: /li	b/pico_i2c_lcd_test @ 63÷1	- 🗆 X
Thonny - Raspberry Pi Pico :: /li File Edit View Run Tools He		
File Edit View Run Tools He		
File Edit View Run Tools He	dp - Save to Raspberry Pi Pico X	
File Edit View Run Tools He		
File Edit View Run Tools He File Market Computer C: \ Users \ muco	lp Save to Raspberry Pi Pico Raspberry Pi Pico	
File Edit View Run Tools He	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He File Market Computer C: \ Users \ muco	Ap Save to Raspberry Pi Pico X Raspberry Pi Pico = Name Size (bytes)	
File Edit View Run Tools He Files X This computer C: \ Users \ muco D J 3D Objects Raspberry Pi Pico v J Ibb	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He File Xi I III IIII IIIIIIIIIIIIIIIIIIIIIIII	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He	4p Save to Raspberry PI Pico X Raspberry Pi Pico = / ibb Name Size (bytes) log Icd_api.py 6996	
File Edit View Run Tools He Files X This computer C: \ Users \ muco D J 3D Objects Rapberry Pi Pico V J Ibb © Icd_api.py © pico_i2c_Icd.py © adc.py © blink.py	Ap Save to Respberry Pi Pico × Respberry Pi Pico • / lib Name Size (bytes) @ Icd_api.py 6996 @ pico_i2c_icd.py 3163	×
File Edit View Run Tools He	Ap	×
File Edit View Run Tools He	Ap Save to Respberry Pi Pico × Respberry Pi Pico • / lib Name Size (bytes) @ Icd_api.py 6996 @ pico_i2c_icd.py 3163	×
File Edit View Run Tools He	4p Save to Respberry Pi Pico × Respberry Pi Pico / // ib Name Size (bytes) <pre></pre>	×
File Edit View Run Tools He	Ap	×
File Edit View Run Tools He	4p Save to Respberry Pi Pico × Respberry Pi Pico / // ib Name Size (bytes) <pre></pre>	×
File Edit View Run Tools He	4p Save to Respberry Pi Pico × Respberry Pi Pico / // ib Name Size (bytes) <pre></pre>	×
File Edit View Run Tools He	4p Save to Respberry Pi Pico × Respberry Pi Pico / // ib Name Size (bytes) <pre></pre>	×

Now that you have added libraries, you can open a new file by clicking the "Plus" icon on the top of a new Thonny IDE and write the following codes. When you examine the codes, first added the libra-ries as in other projects, then made the settings of our 2X16 LCD screen and created a new text docu-ment named "save" with the code "file = open (" record.txt "," w ").

In this part, you have to choose the location where want to save the file, then type the file name first. After that double click on the "lib" folder.

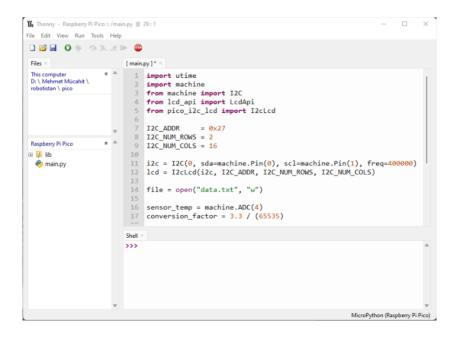
Thonny - C:\Users\muco\D)esktop\lcd_api.py @ 4:5		-		×
File Edit View Run Tools	Help				
🛉 🖄 📩 🜔					
Files X	😨 Save to Raspberry Pi Pico	×			
This computer C: \ Users \ muco	Raspberry Pi Pico				Î
Image: Strange Stra	Name 2 Size (b)	-			
Raspberry Pi Pico	₩ lib adc.py	683	ompatible o the LCD		
lib adc.py	🗢 blink.py	192	o the LCD	, and	1
🕏 blink.py		546 249	ement the	hal_	×
 hirsiz.py temp.py 	- company		the avrl	ib lc	d.
- compipy		- 11			
		- 11	4		
	1	×.	-		•
	File name: Icd_api.py OK C	ancel	-		
			1		
	÷.		icroPython (Ras		Ļ

Then click the "OK" button to save it to the folder.

Thonny - C:\Users\muco		- 🗆 X
Files X This computer C: \ Users \ muco		1
▷ 🖗 3D Objects	Raspberry Pi Pico / lib Name Size (bytes)	
Raspberry Pi Pico lib dc.py blink.py blink.py hirsiz.py temp.py	Vane (vyte)	pmpatible charac b the LCD, and r ement the hal_xx the avrlib lcd.
	File name kd_api.py OK Cancel	MicroPython (Raspberry Pi Pico)

```
1 import utime
2 import machine
3 from machine import I2C
4 from lcd api import LcdApi
5 from pico i2c lcd import I2cLcd
6
7 I2C_ADDR
              = 0x27
8 12C NUM ROWS = 2
9 12C NUM COLS = 16
10
11 i2c = I2C(0, sda=machine.Pin(0), scl=machine.Pin(1), freq=400000)
12 lcd = I2cLcd(i2c, I2C ADDR, I2C NUM ROWS, I2C NUM COLS)
13
14 dosya = open("data.txt", "w")
15
16 sensor temp = machine.ADC(4)
17 conversion_factor = 3.3 / (65535)
18
19 while True:
20
     lcd.clear()
     reading = sensor temp.read u16() * conversion factor
21
22
     temp = 27 - (reading - 0.706)/0.001721
     lcd.putstr("Temp: ")
23
24
     lcd.move to(0,1)
25
     lcd.putstr(str(temp))
26
     dosya.write(str(temp) + "\n")
     dosya.flush()
27
28
      utime.sleep(2)
```

Now save these codes to your Pico and run them, then press the "Run" icon at the top.



It asks where you want to save the code. For the code to always run, you need to click on the "Raspberry Pi Pico" option and save it to our Pico.

Files ×			led> * ×				
This computer D: \ Mehmet Mücahit \ oobotistan \ pico	= ^	14 15 16 17 18	sensor_temp	"data.txt", "w") = machine.ADC(4) actor = 3.3 / (6	1		
		18	while True:				
	-	20	lcd.clea	ir()			
aspberry Pi Pico		21		- concon town no	und 16() * conversion_fa	actor	
Ib		22	eemp .	Where to save to?	× /0.001721		
		23	lcd.p				
		24	lcd.m				
		25 26	lcd.p	This computer			
		26	file. file.				
		27	utime				
		29	ucrite	Raspberry Pi Pico			
		30		Respondy PTPRO			
		Shell >					
		Shell >					

Type "weather_monitor.py" as the file name and click the "OK" button. Now, you will be able to see the temperature data on your LCD screen and at the same time this data will be recorded in a text document.

e Edit View Run Tools	Help	
) 💕 🖩 🛛 🕸 🧆 🤉	e 🕨 🤓	
Files ×	The Save to Raspberry Pi Pico	×
This computer D: \ Mehmet Mücahit \	Raspberry Pi Pico	= A
obotistan \ pico	Name	Size (bytes)
	🔋 lib	
aspberry Pi Pico		ictor
🔊 lib		
		*
	File name: weather_monitor.py	OK Cancel
	Shell ×	
	>>>	
	- MAR	MicroPython (Raspberry Pi Pic

Now wait for some data to collect, then click the "Stop" icon at the top to stop the code from running.

<pre>de Edit View Run Tools Help</pre>	🆬 Thonny - Raspberry Pi Pico			\times
Files [weather_monitor.py]*] This compater Bo VMAmed Muschek rebotitan \pico 1 import utime 1 import machine from machine import 12C 4 from lcd_api import Lcdpi from moci_i2c.lcd import 12Lcld Respberry RPico * Weather_monitor.py 8 12C_NUM_ROWS = 2 12C_NUM_COLS = 16 I 12C_LOUR_ROWS = 16 I 12C_ICd(12c, 12C_ADDR, 12C_NUM_ROWS, 12C_NUM_COLS) I 12c = 12C(d(12c, 12C_ADDR, 12C_NUM_ROWS, 12C_NUM_COLS) I 15 sensor_temp = machine.ADC(4) I/ conversion_factor = 3.3 / (65535)				
This computer D:\\Memma Muchat\ D:\\Memma Muchat\ roomschime Muchat\	🗋 💕 🖩 🛛 🐐 👁 🤉	R I		
D: \Memier Muchet \ D: \Me	Files ×		[weather_monitor.py] ×	
15 16 senson_temp = machine.ADC(4) 17 conversion_factor = 3.3 / (65535) Shell =	D: \ Mehmet Micahit \ robotistan \ pico	· ·	<pre>2 import machine 3 from machine import I2C 4 from 1cd_api import L2C 6 7 IZC_ADDR = 0x27 1ZC_NUM_ROWS = 2 9 IZC_NUM_COLS = 16 10 12c = I2C(0, sda=machine.Pin(0), scl=machine.Pin(1), freq=400000) 12 lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS, I2C_NUM_COLS) 13</pre>	
			<pre>16 sensor_temp = machine.ADC(4) 17 conversion_factor = 3.3 / (65535)</pre>	
*			Shell ×	
· ·			»»	^
MicroPython (Rasoberry Pi Pic		Ŧ		

Write the following codes in the Shell section at the bottom so that you can see the data have saved.

	c (f cl	ather,monitor.py @ 14:23 — — — X	
Files ×		[weather_monitor.py] ×	
This computer D: \ Mehmet Mücahit \ robotistan \ pico	* ^ ~	<pre>1 import utime 2 import machine import I2C 4 from machine import L2CApi 5 from pico_i2c_lcd import L2cLcd 6 7 I2C_ADDR = 0x27 8 I2C.NUM ROWS = 2</pre>	
Raspberry Pi Pico	= ^	9 I2C_NUM_COLS = 16	
 Ib data.txt weather_monitor.py 		<pre>10 11 i2c = I2C(0, sda=machine.Pin(0), scl=machine.Pin(1), freq=400000) 12 lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS, I2C_NUM_COLS) 13 14 file = open("data.txt", "w") 15 sensor_temp = machine.ADC(4) 17 conversion_factor = 3.3 / (65535) 10 10 11 11 11 11 11 11 11 11 11 11 11</pre>	
		Shel ≈ >>> file = open("data.txt") 4	h
	V	MicroPython (Raspberry Pi Picc	" D)

<pre>% how Part Sector Sector (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)</pre>			
Control			rr_monitor.py © 14:23 — □
Discreption Discr			•
D: \Membra MuchA\ indefinition \pice B: \Pice A = \Pic	Files ×		weather_monitor.py] ×
<pre>Stell = open("data.txt") >>> print(file.read()) *</pre>	D: \ Mehmet Mücahit \ robotistan \ pico Raspberry Pi Pico @ Ji Bb @ data.bt	* ^ *	<pre>2 import machine 2 import machine import IZC 4 from lcd_spi import L2Cd 5 from lcd_spi import L2Cdd 5 from lcd_spi import L2Cdd 5 from lcd_spi import L2Cdd 5 from lcd_spi import L2Cdd 5 fize_MM_COLS = 16 1 iZc_MM_COLS = 16 1 iZc_E IZC(0; sdamachine.Pin(0), scl=machine.Pin(1), freq=400000) 1 cd IZclcd(izc, IZC_ADOR, IZC_MMM_ROKS, IZC_MMM_COLS) 1 file = open("data.txt", "w") 1 file = open("data.txt", "w") 1 sensor_temp = machine.ADC(4) 1 conversion_factor = 3.3 (< (5535)) 1 import = 100000000000000000000000000000000000</pre>
<pre>>>> print(file.read()) *</pre>			hell ×
		Ŧ	MicroPython (Raspberry Pi Pi

After typing the "print (file.read ())" code, you can see the saved data as in the photo below.

			\times
The Thonny - Raspberry Pi Pico			
File Edit View Run Tools			
		►	
Files ×		[weather_monitor.py] ×	
This computer D: \ Mehmet Mücahit \ robotistan \ pico		1 import utime 2 import machine import 12C 4 from Iachine import 12C 5 from pico_i2c_lcd import I2clcd 6 7 I2C_ADOR = 0x27 8 I2C_IMM ROS = 2	
haspoeny Preico	-	9 I2C_NUM_COLS = 16	
 ■ jib iib data.bt 		<pre>10 11 i2c = I2C(0, sda=machine.Pin(0), scl=machine.Pin(1), freq=40000 12 lcd = I2clcd(i2c, I2C_ADDR, I2C_NUM_ROWS, I2C_NUM_COLS) 13 14 file = open("data.txt", "w")</pre>	10)
		<pre>sensor_temp = machine.ADC(4) conversion_factor = 3.3 / (65535) </pre>	
		Shell ×	
		<pre>>>> file = open("data.txt") >>> print(file.read())</pre>	
		21.42667 21.42667 21.42667 21.42667	1
	Ŧ	22.36296	v Pi Pico)
_	Ŧ	22.36296 MicroPython (Raspber	
Ĩn Thonny - Raspberry Pi Pico File Edit View Run Tools □ 2 3 1 0 1	Help	22.36296 MicroPython (Rapher ther_monitor.py @ 14123 —	
File Edit View Run Tools	Help	22.36296 MicroPython (Rapter thes_monitor.py © 14:23	
File Edit View Run Tools	Help	22.36296 MicroPython (Rapher ther_monitor.py @ 14123 —	
File Edit View Run Tools Edit View Run Tools Files Files Files Files Files Raspberry Pi Pico Files iles Files Files Fi	Help	22.36296 MorePyton (Rapter More	
File Edit View Run Tools	Help	<pre>22.36296 MonoPython Respect ther_monotargy © 14:23</pre>	×
File Edit View Run Tools File Help	22.36296 MorePython (Rapper thes_member.py) = 14:23	×	
File Edit View Run Tools File Help	<pre>22.36296 MonoPython Respect ther_monotargy © 14:23</pre>	×	
File Edit View Run Tools File Help	<pre>22.36296 MonoPython Respect the monotory = 14:23</pre>	×	

Burglar Alarm







maker.robotistan.com





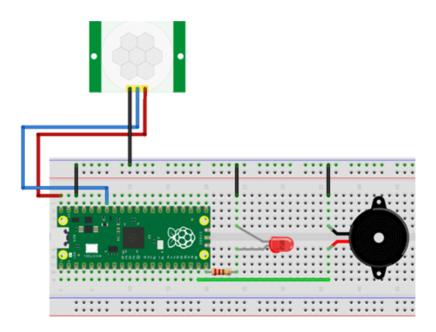


In this project, you will make a burglar alarm using PIR sensor, LED and buzzer.

Necessary materials:

- Raspberry Pi Pico
- Breadboard
- PIR Sensor
- Buzzer
- LED
- Male Male Jumper
- Male Female Jumper
- 220 Ω or 330 Ω Resistor

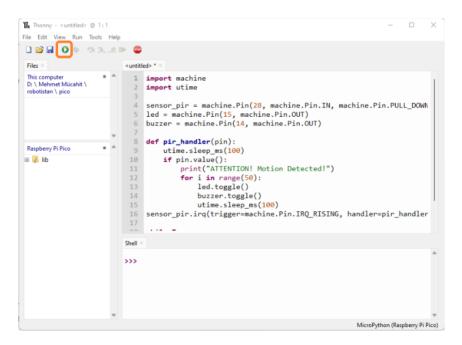
First, build your circuit according to the diagram below.



Now open a new file in Thonny IDE and write the following codes. . When you examine the codes, you first added the libraries and then defined the pins to which you connected the PIR sensor, Buzzer and LED. You created a function with "def pir_handler (pin)". When the motion is detected with the if block in the function, "ATTENTION! Motion Detected! " printing, buzzer and LED are wor-king.

```
1 import machine
 2 import utime
 3
 4 sensor pir = machine.Pin(28, machine.Pin.IN, machine.Pin.PULL DOWN)
 5 led = machine.Pin(15, machine.Pin.OUT)
 6 buzzer = machine.Pin(14, machine.Pin.OUT)
7
8 def pir handler(pin):
9
      utime.sleep ms(100)
      if pin.value():
10
          print("ATTENTION! Motion Detected!")
11
12
          for i in range(50):
13
              led.toggle()
14
              buzzer.toggle()
15
              utime.sleep ms(100)
16 sensor pir.irq(trigger=machine.Pin.IRQ RISING, handler=pir handler)
17
18 while True:
19
     led.toggle()
20
      utime.sleep(5)
```

Now save these codes to your Pico and run them, then press the "Run" icon at the top.



It asks where you want to save the code. For the code to always run, you need to click on the "Raspberry Pi Pico" option and save it to our Pico.

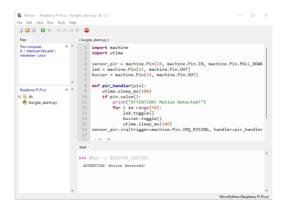
R Thonny - <untitled> @ ile Edit View Run Te</untitled>					- 0
D 😂 🖬 🛛 🕸 🕫	(ae.)	▶ 😳			
Files ×		<untitled> * ></untitled>			
This computer D: \ Mehmet Mücahit \ robotistan \ pico	•	2 imp 3 4 sen: 5 led 6 buz: 7	= machin zer = mac	machine.Pin(28, ne.Pin(15, machine chine.Pin(14, mach	
Raspberry Pi Pico		8 def		dler(pin): Where to save to?	×
a 🤑 lib		10 11 12 13 14 15	if pi p f	This computer	Detected!")
			sor_pi	Raspberry Pi Pico	.n.IRQ_RISING, handler=pir_handler
		Shell ×	_		

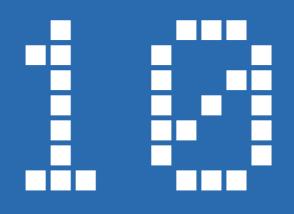
	-				

Write "burglar_alarm.py" as the file name and click on the "OK" button and now, in case of any situation that triggers our motion sensor, the buzzer will sound and our LED will light.

his computer t \ Mehmet Mücahit \ bobotistan \ pico	Save to Raspberry Pi Pico Raspberry Pi Pico Name Rasp Bit Pico Raspberry Pi o Raspberry Pi Pico Raspberry Pi Pico Raspberry Pi Pico Raspberry Pico Raspberry Pi Pi		Size (byt	× = * +.P	'in.PULL_DOW
aspberry Pi Pico	Raspberry Pi Pico Name		Size (byt	= ^	in.PULL_DOW
r \ Mehmet Mücahit \ obotistan \ pico aspberry Pi Pico =	Name		Size (byt	(1)	in.PULL_DOW
aspberry Pi Pico	Name		Size (byt	(1)	in.PULL_DOW
aspberry Pi Pico			Size (byt	es) 1.P.	in.PULL_DOW
	🤾 lib			1.0	10.POLL_DOM
	File name: burglar_alarm	чру	OK C	ancel .er	=pir_handle
	Shell ×				
	>>> %Run -c \$EI	DITOR_CONTENT			
	>>>				
		.17 on 2021-09-02 for more informat	2; Raspberry Pi Pico	with RP20	040

In the Shell window at the bottom, when motion is detected, "ATTENTION! Motion Detected!" text will appear.





Using of Distance Sensor







FORUM





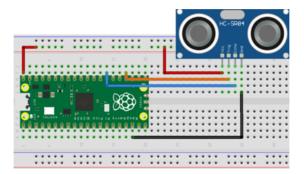
youtube.com/robotistan

In this project, you will measure distance with HC-SR04 distance sensor.

Necessary materials:

- Raspberry Pi Pico
- Breadboard
- HC-SR04 Distance Sensor
- Male Male Jumper

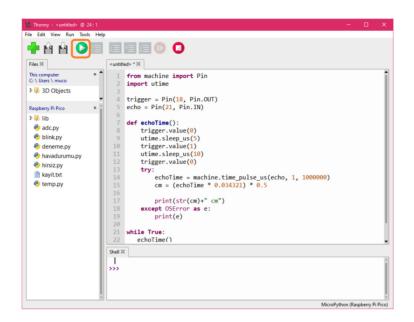
First, create your circuit on the breadboard according to the diagram below.



Now open a new file in Thonny IDE and write your code. When you examine the codes, you defined the libraries in the first two lines, and the pins to which the distance sensor is connected. Also wrote the distance measurement function with the function "def echoTime ():". You have made the distance measurement function, which defined with the while loop, run every 5 seconds.

```
1 from machine import Pin
 2 import utime
 4 trigger = Pin(18, Pin.OUT)
 5 echo = Pin(21, Pin.IN)
 6
 7 def echoTime():
 8
     trigger.value(0)
 9
     utime.sleep us(5)
10
     trigger.value(1)
11
     utime.sleep us(10)
12
     trigger.value(0)
13 try:
14
          echoTime = machine.time_pulse_us(echo, 1, 1000000)
         cm = (echoTime * 0.034321) * 0.5
15
16
17
         print(str(cm)+" cm")
18
      except OSError as e:
19
          print(e)
20
21 while True:
22
   echoTime()
23
     utime.sleep(5)
```

Now save these codes to your Pico and run them, then press the "Run" icon at the top.



It asks where you want to save the code. For the code to always run, you need to click on the "Raspberry Pi Pico" option and save it to our Pico.

The Thonny - <untitled> @ 2 File Edit View Run Tool</untitled>			-		×
🕂 🖻 🖄 🜔					
Files 🛛		<untitled> * X</untitled>			
This computer C: \ Users \ muco	• 1	1 from machine import Pin 2 import utime			Î
B 3D Objects Raspberry Pi Pico		3 4 trigger = Pin(18, Pin.OUT) 5 echo = Pin(21, Pin.IN)			
Rasperty Price P i i ib adc.py blink.py blink.py blink.py chavadurum.py havadurum.py havadurum.py havadurum.py kayit.tt for temp.py		<pre>def echoTime(): trigger.value(0) utime.slue(0) trigger. (M. Where is save to? trigger. (M. Where is save to? trigger.) trigger.</pre>			
	Ţ	Micro	ython (Ra	oberry P	i Pico)

	- 0
e 🕨 👳	
The Save to Raspberry Pi Pico	×
Raspberry Pi Pico	= ^
Name	Size (bytes)
💫 lib	
	▼ 100000)
File name: distance.py	OK Cancel
Shell ×	
>>>	
	Raspberry Pi Pico Name Pile lib File name: distance.py Shell ×

Write "distance.py" as the file name and click the "OK" button.

You can now see the distance in the Shell section of the Thonny IDE.

The Thonny - Raspberry Pi P	Pico :: /dist	tance.py @ 15:3	2 –	\Box \times
File Edit View Run To	ols Help	0		
🗋 💕 🖩 🗿 🌞 🤝	36 (distance.py]	×	
This computer D: \ Mehmet Mücahit \ robotistan \ pico	± ^	1 from 2 impo 3 4 trig 5 echo 6 7 def	<pre>machine import Pin rt utime ger = Pin(18, Pin.OUT) = Pin(21, Pin.IN) echoTime(): trigger.value(0)</pre>	
Raspbery Pi Pico	*	9 10 11 12	<pre>trigger.value(0) trime.sleep_us(5) trigger.value(1) utime.sleep_us(10) trigger.value(0) try: echoTime = machine.time_pulse_us(echo, 1, 1000000) cm = (echoTime * 0.034321) * 0.5 print(str(cm)+" cm") core</pre>	
	Ţ	Shell × >>> %Run 170.7984 2.985927 10.03889 4.701977 169.4771	cm cm cm	-
		-	MicroPython (Raspt	berry Pi Pico)





youtube.com/robotistan





Robotistan Elektronik Ticaret A.Ş.

Mehmet Mücahit KAYA(Content) - Mehmet AKÇALI - Yasin TAŞCIOĞLU (Editor) - Mehmet Nasır KARAER (Graphic) info@robotistan.com - www.robotistan.com Tel: 0850 766 0 425