

MultiWingSpan

Home Programming Web Design Computer Science Twisting Puzzles Arduino BBC micro:bit

BBC micro:bit Encoding Morse Code

Introduction

This project is the beginning of something more complicated. You will need to have the serial port driver installed on the computer you are using and have access to a terminal emulator to use REPL with the micro:bit.

The program will then automatically encode text into a pattern of dots and dashes in Morse Code. These are then flashed out on the LED matrix as well as being printed for your viewing pleasure.

```
from microbit import *

# 'constants' for delays all derived from dot length
dotlength = 250
dashlength = dotlength * 3
interelement = dotlength
interletter = dotlength * 2

# images for displaying dots and dashes
dot_img = Image('00000:00000:00900:00000:00000:')
dash_img = Image('00000:00000:99999:00000:00000:')

# Dictionary
morse = {
    "A": ".-.",
    "B": "-... ",
    "C": "-.-. ",
    "D": "-.. ",
    "E": ". ",
    "F": ".-.- ",
    "G": "--. ",
    "H": ".... ",
    "I": ". . ",
    "J": "-.-.- ",
    "K": "-.-. ",
    "L": ".-.. ",
    "M": "-- ",
    "N": "-. ",
    "O": "--- ",
    "P": "-.-. ",
    "Q": "--.- ",
    "R": ".-.- ",
    "S": "... ",
    "T": "-. ",
    "U": "...- ",
    "V": "...- ",
    "W": "-.-. ",
    "X": "-.-.- ",
    "Y": "-.-.- ",
    "Z": "--.-. "
}

# function to convert string to pattern of dots and spaces
def EncodeMorse(message):
    m = message.upper()
    enc = ""
    for c in m:
        enc = enc + morse.get(c, " ")
        if morse.get(c, " ") != " ":
            enc = enc + " "
    return enc

# function to flash out a Morse pattern on the matrix
def FlashMorse(pattern):
    for c in pattern:
        if c == ".":
            display.show(dot_img)
            sleep(dotlength)
            display.clear()
            sleep(interelement)
        elif c == "-":
            display.show(dash_img)
            sleep(dashlength)
            display.clear()
            sleep(interelement)
        elif c == " ":
            sleep(interletter)
    return

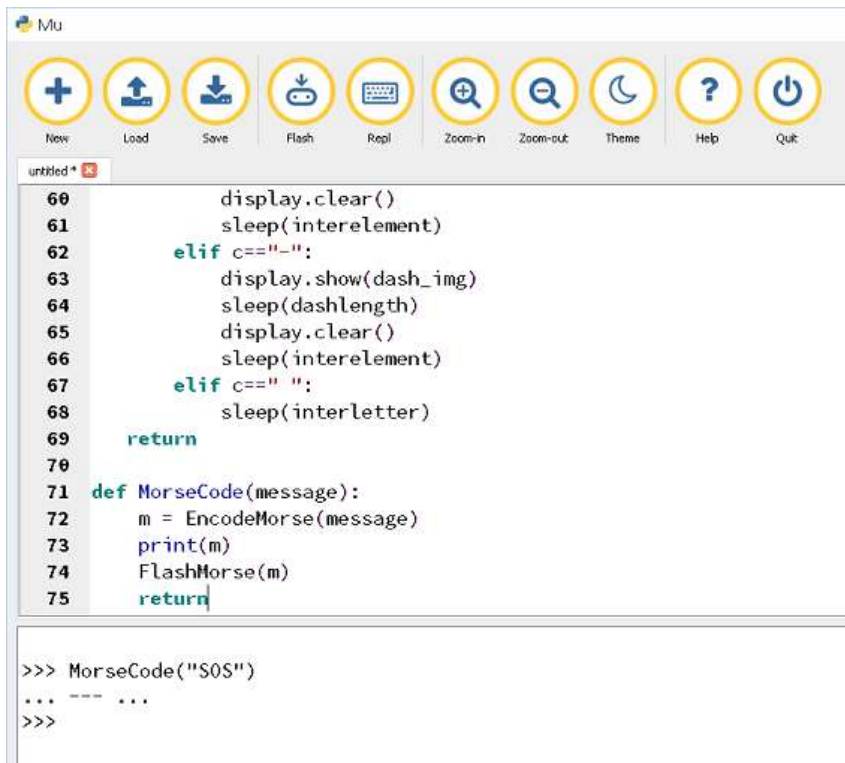
# helper function to encode and flash in one go
def MorseCode(message):
    m = EncodeMorse(message)
    print(m)
    FlashMorse(m)
    return
```

BBC Microbit

- + **Block Editor - The Basics**
- + **Block Editor - Components**
- + **Kodu - micro:bit Worlds**
- + **JavaScript Blocks**
- + **JavaScript Blocks - Exercises**
- + **Blocks - Bit:Bot**
- + **Blocks - Bit:Commander**
- + **MicroPython - Starting Off**
- **MicroPython - Examples**
 - ✱ **Dice Rolling**
 - ✱ **Shut The Matrix**
 - ✱ **Encoding Morse Code**
 - ✱ **Encoding Ciphers**
 - ✱ **Drawing A Maze**
 - ✱ **Scrolling Race Track**
 - ✱ **Vertical Scroller**
 - ✱ **Concentration Game**
 - ✱ **Text Entry - Accelerometer**
 - ✱ **Charlie's Python Game**
 - ✱ **Lights Out Game**
 - ✱ **Sam's French Number Game**
 - ✱ **Bryn's Concentrated Clocks**
 - ✱ **Lattice Paths**
 - ✱ **Knight Moves**
- + **MicroPython - Components**
- + **MicroPython - Breakout Boards**
- + **MicroPython - Exercises**
- + **MicroPython - Pi Accessories**
- + **MicroPython - Bit:Bot**
- + **MicroPython - Bit:Commander**
- + **MicroPython - Projects**
- + **MicroPython - Visual Basic**
- + **Other - Odds & Ends**



When you have flashed the program to the micro:bit, open up the REPL window and, at the prompt, type a call to the MorseCode function like this,



The screenshot shows the Mu editor interface. At the top is a toolbar with icons for New, Load, Save, Flash, Repl, Zoom-in, Zoom-out, Theme, Help, and Quit. Below the toolbar is a code editor with a file named 'untitled+'. The code is as follows:

```
60     display.clear()
61     sleep(interelement)
62     elif c=="-":
63         display.show(dash_img)
64         sleep(dashlength)
65         display.clear()
66         sleep(interelement)
67     elif c==" ":
68         sleep(interletter)
69     return
70
71 def MorseCode(message):
72     m = EncodeMorse(message)
73     print(m)
74     FlashMorse(m)
75     return
```

Below the code editor is a REPL window. It shows the command `>>> MorseCode("SOS")` being entered, followed by the output `... --- ...` and another `>>>` prompt.

Challenges

1. This program really needs a buzzer to finish everything off. Connect either a buzzer or crocodile clip some headphones to pin 0. Adjust the program so that it buzzes a suitable length note for the dashes and dots.
2. The morse code dictionary only has upper case letters. Look around online and you can find the standard patterns for numbers and other characters. Extend the program so that these are included.
3. Notice the way that you can make calls to the functions you have written when using the REPL window. Have a think about how you might combine the programming techniques you have already used to make use of this feature somehow.