Christmas Project 2023

Contents

- Santas Grotto or Penguin House
- Raspberry Pi Pico H running CircuitPython 8.2.6.
- Circuit board with:
    - 1 x Buzzer
    - 3 x LEDs
    - 1 x 8 NeoPixel ring

Raspberry Pi Pico and Key files

Connect the Raspberry Pi Pico to the Computer with a USB cable. The Pico is running CircuitPython and once plugged in will show up as a USB device and start executing the Python code. With default settings, the code will go into demo mode which will animate the LEDS, animate the NeoPixels and play music.

Any time you change a file on the Pico and save it, CircuitPython will reload the program and start from scratch.

There are two important files that can be modified to control what the Pico does. The first is the file `config.py`. Open `config.py` in notepad. This file contains configuration data that tells the Pico which pins on the circuit board the LEDs, NeoPixels and Buzzer are connected to. It also contains configuration data to control which code gets executed. In particular these settings are useful.

Setting	Values
PLAY_SONGS	`True` – The buzzer will play sound. `False` – The buzzer will be silent
AUDIO_VOLUME	A value from 0 (silent) to 0.1 (maximum volume).
ANIMATES_LEDS	`True` – The Red, Green and Yellow LEDs will animate. `False` – The Red, Green and Yellow LEDS will be off.
ANIMATE_PIXELS	`True` – The NeoPixels will animate. `False` – The NeoPixels will be off.
PIXELS_BRIGHTNESS	A value from 0 (off) to 1.0 (maximum brightness).
DEMO_MODE	`True` – Run the demo code. `False` – Run your own code.

Change the value of PLAY_SONGS, AUDIO_VOLUME, ANIMATE_LEDS, ANIMATE_PIXELS and PIXEL_BRIGHTNESS to see what effect this has.

Christmas Project 2023

Demo code

The code that controls the demo mode is in the file `demo.py`. Open the file and look at the code. It is split into 3 sections, one each for the music, the LEDs and the NeoPixels.

Music

This section creates a sequence of all the Christmas songs that loops around. The four songs are:

1. Jingle Bells
2. Rudolph the Red Nosed Reindeer
3. When Santa Got Stuck Up the Chimney
4. We Wish You a Merry Christmas

The button can be used to pause and resume the music.

LEDs

Each of the three LEDs have animations setup. The red LED will blink, the green LED will pulse and the Yellow LED will cycle through the sequence of flicker, pulse and blink.

NeoPixels

The pixels will cycle through eleven different animations with each one lasting 5 seconds.

Writing your own code

Turning off the demo mode (setting DEMO_MODE = False in `config.py`) means you can write your own code to control the LEDS, NeoPixels and Buzzer. Open the file `code.py` in notepad and inside that file you will see a function called `init()`. This is where you will copy the code from `demo.py`.

The `init()` function already contains a single animation for the yellow LED and a single animation for the NeoPixels. Adjust the settings for each of the animations to see how it changes the effect. For example, change the speed of the comet animation to 0.5 and change bounce to `False`. Change the color.

Copy across the code for the other LED and NeoPixel animations that you want to experiment with. Find the settings that you like the most.

Music

Making your own song is easy. Look at the code in the file Christmas_songs.py for examples of how to do it. Each song is a list of notes to play. Each note consists of 2 parts, a note/beat (duration) pair. The note can optionally be given an octave. Setting the octave sets that octave for all future notes. To play a simple scale of C can be done by adding the following code to the `init()` method.

```
notes = [
    "C4:1", "D:1", "E:1", "F:1", "G:1", "A:1", "B:1", "C5:1",
    "B4:1", "A:1", "G:1", "F:1", "E:1", "D:1", "C:1"]
song = Song(audio, decode_song(notes), 0.2)
framework.music = SongSequence(song, loop=True)
```