```python
from math import floor
from modules.runify import runify
from threading import Thread
from pybricks.media.ev3dev import Font
from pybricks.parameters import Button, Color
from pybricks.tools import StopWatch, wait


# Class to control running runs
class menu:
    index = 0
    page = 0
    refresh_time = 100
    max_items = 4

    def __init__(self, config, volume):
        # If sound gets too annoying
        self.ev3 = config.ev3
        self.ev3.speaker.set_volume(volume)

        # Gets configuration
        self.config = config

        # Gets menu data from config
        tempMenu = config.menu
        self.pages = config.menu["pages"]
        del config.menu["pages"]

        self.menu = {}

        self.menu["runs"] = tempMenu["runs"]
        self.menu["left"] = tempMenu["left"]

        for page in self.pages:
            if page != "runs" and page != "left":
                temp = [runify(func, self.config)
                        for func in tempMenu[page][1]]
                self.menu[page] = [tempMenu[page][0], temp]  # type: ignore

        # Sets up font for menu
        font = Font("Terminal", 16, monospace=True)
        self.ev3.screen.set_font(font)

        # Change status light to standby
        self.ev3.light.on(Color.RED)

        # If battery level too low, give a longer beep

        if self.ev3.battery.voltage() < 8100:
            Thread(target=self.ev3.speaker.beep, args=[1500, 2000]).start()
            # self.ev3.speaker.beep(1500, 2000)
        else:
            Thread(target=self.ev3.speaker.beep, args=[1000, 100]).start()
            # self.ev3.speaker.beep(frequency=1000, duration=100)

    # Main control loop
    # Handles button presses
    def update(self):
        # Makes sure index is within bounds of menu
        self.page = self.wrap_index(self.page, self.pages)
        self.index = self.wrap_index(
            self.index, self.menu[self.pages[self.page]][0])

        # Displays all data
        self.displayMenu(self.index, self.page)

        # Makes sure no button is pressed twice
```

```
68            wait(self.refresh_time)
69            self.refresh_time = 100
70
71            # Gets buttons that are pressed
72            button = self.ev3.buttons.pressed()
73
74            # Makes sure only one button is pressed
75            if len(button) == 1:
76                # If middle button, run the run selected
77                if Button.CENTER in button:
78                    self.run(self.menu[self.pages[self.page]]
79                             [1][self.index])
80                    self.index += 1  # At end of run, move to next run
81
82                # Moves up in the menu
83                elif Button.UP in button:
84                    self.index -= 1
85                    self.refresh_time = 400
86
87                # Moves down in menu
88                elif Button.DOWN in button:
89                    self.index += 1
90                    self.refresh_time = 400
91
92                # Each run has a corresponding function that can be run through the
93                # left button
94                elif Button.LEFT in button:
95                    if self.menu["left"][self.index] != None:
96                        self.menu["left"][self.index]()
97                    else:
98                        print("Nothing assigned")
99                    self.refresh_time = 400
100
101               # Switch pages
102               elif Button.RIGHT in button:
103                   self.page += 1
104                   self.index = 0
105                   self.refresh_time = 400
106
107           # If no buttons are press, check if runButton exists and is pressed
108           # If true, run the run too
109           elif self.config.runButton != None and self.config.runButton.pressed() == True:
110               self.run(self.menu[self.pages[self.page]]
111                        [1][self.index])
112               self.index += 1  # At end of run, move to next run
113
114       def wrap_index(self, idx, theList):
115           if idx >= len(theList):
116               idx = 0
117           elif idx < 0:
118               idx = len(theList)-1
119           return idx
120
121       # Displays all information on screen
122       def displayMenu(self, curr_index, pageIdx):
123           self.ev3.screen.clear()
124           count = 0
125           if floor(curr_index / self.max_items) * self.max_items > 0:
126               self.ev3.screen.print("  ...")
127
128           for item in self.menu[self.pages[pageIdx]][0]:
129               count += 1
130               if count <= floor(curr_index / self.max_items) * self.max_items:
131                   continue
132
133               if self.menu[self.pages[pageIdx]][0].index(item) == curr_index:
134                   self.ev3.screen.print(">", item)
```

```
135                 else:
136                     self.ev3.screen.print(" ", item)
137 
138                 if count >= floor(curr_index / self.max_items) * self.max_items + self.
                    max_items:
139                     self.ev3.screen.print("  ...")
140                     break
141 
142             self.ev3.screen.print(
143                 self.config.name, ":", self.ev3.battery.voltage(), end="")
144 
145         def displayInfo(self):
146             self.ev3.screen.clear()
147             for i in self.config.display:
148                 self.ev3.screen.print(i())
149 
150         def infoLoop(self):
151             while True:
152                 self.displayInfo()
153                 self.config.timer.wait(100)
154 
155         # Runs given run
156         def run(self, func):
157             self.ev3.speaker.beep(frequency=1000, duration=250)
158 
159             self.config.state.setState(self.config.state.running)
160 
161             # Start run in another thread (in parallel)
162             func.start()
163 
164             # Update status light
165             self.ev3.light.on(Color.GREEN)
166 
167             # Wait for 2 seconds or until run button is released
168             timer = StopWatch()
169             while timer.time() < 2000 and (self.config.runButton != None and self.config.
                runButton.pressed() ==
170                                            True) or Button.CENTER in self.ev3.buttons.pressed
                                               ():
171                 wait(20)
172 
173             # Wait until run finishes or is stopped via run button
174             while self.config.state.getState() != 1:
175                 if (self.config.runButton != None and self.config.runButton.pressed() ==
176                         True) or Button.CENTER in self.ev3.buttons.pressed():
177                     self.config.state.setState(self.config.state.stop)
178 
179                 self.displayInfo()
180                 wait(200)
181 
182             # Reset
183             self.config.stop()
184             self.ev3.speaker.beep(frequency=1000, duration=250)
185             self.ev3.light.on(Color.RED)
186             if self.pages[self.page] == "runs":
187                 print(self.menu[self.pages[self.page]][0]
188                       [self.index], "Took:", timer.time(), "ms")
189             self.config.state.setState(self.config.state.standby)
190 
```