

```

1  from pybricks.hubs import EV3Brick
2  # from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
3  #                                  InfraredSensor, UltrasonicSensor, GyroSensor)
4  from pybricks.parameters import Port, Stop, Direction, Button, Color
5  from pybricks.tools import wait, StopWatch, DataLog
6  from pybricks.robotics import DriveBase
7  from pybricks.media.ev3dev import Font, SoundFile, ImageFile
8
9  from modules.components.drivebase import DriveBaseFull
10 from modules.components.forklift import Forklift
11 from modules.components.lightSensor import LightSensor
12 from modules.components.runButton import RunButton
13 from modules.components.gyro import GyroSensor
14 from modules.components.tools import RunState, Timer
15 from modules.components.motor import Motor
16
17 from regionals.windmillRun import windmillRun
18 from regionals.powerPlantRun import powerPlantRun
19 from regionals.solarRun import solarRun
20 from regionals.oilRun import oilRun
21 from regionals.hydroRun import hydroRun
22 from regionals.toyRun import toyRun
23
24 from os import popen
25
26
27 # Holds information for one or more robots
28 class config:
29     def __init__(self):
30         # Gets hostname to identify robot
31         self.name = popen('hostname').read().strip()
32
33         self.ev3 = EV3Brick()
34         self.state = RunState()
35         self.runButton = None
36         self.menu = {"left": []}
37
38         self.stopList = []
39         self.display = []
40
41         self.timer = Timer(self)
42
43         # Define all robots beneath
44         if self.name == "artemis":
45             self.SPEED_LIST_COUNT = 2000
46             self.ACCELERATION = 250
47             self.STARTSPEED = 50
48             self.TURN_SPEED_MIN = 20
49             self.TURN_SPEED_MAX = 180
50             self.LIGHTCAL_CONF = "artemis.cal"
51
52             self.Lmotor = self.init(
53                 Motor, Port.B, self, Direction.COUNTERCLOCKWISE)
54             self.Rmotor = self.init(
55                 Motor, Port.C, self, Direction.COUNTERCLOCKWISE)
56             self.LMmotor = self.init(Motor, Port.A, self)
57             self.RMmotor = self.init(Motor, Port.D, self)
58
59             # self.runButton = runButton(TouchSensor(Port))
60             self.gyro = self.init(GyroSensor, Port.S1,
61                                   Direction.COUNTERCLOCKWISE)
62             self.Llight = self.init(LightSensor, Port.S3)
63             self.Rlight = self.init(LightSensor, Port.S4)
64
65             self.lift = Forklift(self, self.RMmotor, 11, 40, 8)
66             self.drive = DriveBaseFull(self, self.Lmotor, self.Rmotor, self.gyro,
67                                         56, 104, self.runButton, Llight=self.Llight,

```

```

68         Rlight=self.Rlight)
69
70     self.menu = {
71         "runs": [ ["powerPlantRun", "windmillRun", "solarRun", "oilRun",
72                 "hydroRun", "toyRun"], [powerPlantRun(self), windmillRun(self), solarRun(
73                 self), oilRun(self), hydroRun(self), toyRun(self)] ],
74         "left": [None, None, None, None, None, None],
75         "utility": [ ["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
76                 self.drive.gyroDrift, self.drive.tyreClean]] ,
77         "pages": ["runs", "utility"]
78     }
79
80     self.display = [self.drive.getHead]
81     self.stopList = [self.drive, self.lift, self.LMmotor, self.RMmotor]
82
83     # self.xlift = forklift(Motor(Port.B))
84     # self.ylift = forklift(Motor(Port.C))
85     # self.lift = doubleForklift(self.xlift, self.ylift)
86
87     elif self.name == "apollo":
88         self.SPEED_LIST_COUNT = 2000
89         self.ACCELERATION = 250
90         self.STARTSPEED = 50
91         self.TURN_SPEED_MIN = 20
92         self.TURN_SPEED_MAX = 180
93         self.LIGHTCAL_CONF = "apollo.cal"
94
95         self.Lmotor = self.init(
96             Motor, Port.B, self, Direction.COUNTERCLOCKWISE)
97         self.Rmotor = self.init(
98             Motor, Port.A, self, Direction.COUNTERCLOCKWISE)
99         self.LMmotor = self.init(Motor, Port.C, self)
100        self.RMmotor = self.init(Motor, Port.D, self)
101
102        # self.runButton = runButton(TouchSensor(Port))
103        self.gyro = self.init(GyroSensor, Port.S1,
104                               Direction.COUNTERCLOCKWISE)
105        self.Llight = self.init(LightSensor, Port.S3)
106        self.Rlight = self.init(LightSensor, Port.S4)
107
108        # self.lift = forklift(self, motor(self,
109        #                                   Port.D, gears=[[12, 20], [28, 20], [8,
110        #                                   40]]), 110)
111
112        self.drive = DriveBaseFull(self, self.Lmotor, self.Rmotor, self.gyro,
113                                   56, 104, self.runButton, Llight=self.Llight,
114                                   Rlight=self.Rlight)
115
116        self.menu = {
117            "runs": [ ["powerPlantRun", "windmillRun", "solarRun", "oilRun",
118                    "hydroRun", "toyRun"], [powerPlantRun(self), windmillRun(self), solarRun(
119                    self), oilRun(self), hydroRun(self), toyRun(self)] ],
120            "left": [None, None, None, None, None, None],
121            "utility": [ ["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
122                    self.drive.gyroDrift, self.drive.tyreClean]] ,
123            "pages": ["runs", "utility"]
124        }
125
126        self.display = [self.drive.getHead]
127        self.stopList = [self.drive, self.LMmotor, self.RMmotor]
128
129        # self.xlift = forklift(Motor(Port.B))
130        # self.ylift = forklift(Motor(Port.C))
131        # self.lift = doubleForklift(self.xlift, self.ylift)
132    else:
133        self.ev3.screen.clear()
134        self.ev3.screen.print("Unkown robot\nNo config found")

```

```
126         while True:
127             wait(100)
128
129     def stop(self):
130         for module in self.stopList:
131             module.stop()
132
133     def init(self, type, port, *args, **kwargs):
134         try:
135             return type(port, *args, **kwargs)
136         except:
137             self.ev3.screen.clear()
138             self.ev3.screen.print(type.__name__, "\nOn port", port)
139             self.ev3.speaker.beep(500, 2000)
140             while True:
141                 wait(1000)
142
```