

```

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 = 380
47             self.STARTSPEED = 60
48             self.TURN_SPEED_MIN = 20
49             self.TURN_SPEED_MAX = 200
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),
73             solarRun(self), oilRun(self), hydroRun(self), toyRun(self)]],
74         # "left": [None, None, None, None, None, None],
75         "runs": [["powerPlantRun", "windmillRun", "solarRun", "oilRun", "toyRun"
76             ], [powerPlantRun(self), windmillRun(self), solarRun(self), oilRun(self),
77             toyRun(self)]],
78         "left": [None, None, None, None, None],
79         "utility": [["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
80             self.drive.gyroDrift, self.drive.tyreClean]],
81         "pages": ["runs", "utility"]
82     }
83
84     self.display = [self.drive.getHead]
85     self.stopList = [self.drive, self.lift, self.LMmotor, self.RMmotor]
86
87     # self.xlift = forklift(Motor(Port.B))
88     # self.ylift = forklift(Motor(Port.C))
89     # self.lift = doubleForklift(self.xlift, self.ylift)
90
91     elif self.name == "apollo":
92         self.SPEED_LIST_COUNT = 2000
93         self.ACCELERATION = 380
94         self.STARTSPEED = 60
95         self.TURN_SPEED_MIN = 20
96         self.TURN_SPEED_MAX = 200
97         self.LIGHTCAL_CONF = "apollo.cal"
98
99         self.Lmotor = self.init(
100             Motor, Port.A, self, Direction.COUNTERCLOCKWISE)
101         self.Rmotor = self.init(
102             Motor, Port.B, self, Direction.COUNTERCLOCKWISE)
103         self.LMmotor = self.init(Motor, Port.C, self)
104         self.RMmotor = self.init(Motor, Port.D, self)
105
106         # self.runButton = runButton(TouchSensor(Port))
107         self.gyro = self.init(GyroSensor, Port.S1,
108             Direction.COUNTERCLOCKWISE)
109         self.Llight = self.init(LightSensor, Port.S3)
110         self.Rlight = self.init(LightSensor, Port.S4)
111
112         # self.lift = forklift(self, motor(self,
113             # Port.D, gears=[[12, 20], [28, 20], [8,
114             40]]), 110)
115
116         self.drive = DriveBaseFull(self, self.Lmotor, self.Rmotor, self.gyro,
117             56, 104, self.runButton, Llight=self.Llight,
118             Rlight=self.Rlight)
119
120         self.menu = {
121             # "runs": [["powerPlantRun", "windmillRun", "solarRun", "oilRun",
122                 "hydroRun", "toyRun"], [powerPlantRun(self), windmillRun(self),
123                 solarRun(self), oilRun(self), hydroRun(self), toyRun(self)]],
124             # "left": [None, None, None, None, None, None],
125             "runs": [["powerPlantRun", "windmillRun", "solarRun", "oilRun", "toyRun"
126                 ], [powerPlantRun(self), windmillRun(self), solarRun(self), oilRun(self),
127                 toyRun(self)]],
128             "left": [None, None, None, None, None],
129             "utility": [["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
130                 self.drive.gyroDrift, self.drive.tyreClean]],
131             "pages": ["runs", "utility"]
132         }

```

```
120
121     self.display = [self.drive.getHead]
122     self.stopList = [self.drive, self.LMmotor, self.RMmotor]
123
124     # self.xlift = forklift(Motor(Port.B))
125     # self.ylift = forklift(Motor(Port.C))
126     # self.lift = doubleForklift(self.xlift, self.ylift)
127 else:
128     self.ev3.screen.clear()
129     self.ev3.screen.print("Unkown robot\nNo config found")
130     while True:
131         wait(100)
132
133 def stop(self):
134     for module in self.stopList:
135         module.stop()
136
137 def init(self, type, port, *args, **kwargs):
138     try:
139         return type(port, *args, **kwargs)
140     except:
141         self.ev3.screen.clear()
142         self.ev3.screen.print(type.__name__, "\nOn port", port)
143         self.ev3.speaker.beep(500, 2000)
144         while True:
145             wait(1000)
146
```