```python
from pybricks.hubs import EV3Brick
# from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
#                                  InfraredSensor, UltrasonicSensor, GyroSensor)
from pybricks.parameters import Port, Stop, Direction, Button, Color
from pybricks.tools import wait, StopWatch, DataLog
from pybricks.robotics import DriveBase
from pybricks.media.ev3dev import Font, SoundFile, ImageFile

from modules.components.drivebase import DriveBaseFull
from modules.components.forklift import Forklift
from modules.components.lightSensor import LightSensor
from modules.components.runButton import RunButton
from modules.components.gyro import GyroSensor
from modules.components.tools import RunState, Timer
from modules.components.motor import Motor

from regionals.windmillRun import windmillRun
from regionals.powerPlantRun import powerPlantRun
from regionals.solarRun import solarRun
from regionals.oilRun import oilRun
from regionals.hydroRun import hydroRun
from regionals.toyRun import toyRun

from os import popen


# Holds information for one or more robots
class config:
    def __init__(self):
        # Gets hostname to identify robot
        self.name = popen('hostname').read().strip()

        self.ev3 = EV3Brick()
        self.state = RunState()
        self.runButton = None
        self.menu = {"left": []}

        self.stopList = []
        self.display = []

        self.timer = Timer(self)

        # Define all robots beneath
        if self.name == "artemis":
            self.SPEED_LIST_COUNT = 2000
            self.ACCELERATION = 250
            self.STARTSPEED = 50
            self.TURN_SPEED_MIN = 20
            self.TURN_SPEED_MAX = 180
            self.LIGHTCAL_CONF = "artemis.cal"

            self.Lmotor = self.init(
                Motor, Port.B, self, Direction.COUNTERCLOCKWISE)
            self.Rmotor = self.init(
                Motor, Port.C, self, Direction.COUNTERCLOCKWISE)
            self.LMmotor = self.init(Motor, Port.A, self)
            self.RMmotor = self.init(Motor, Port.D, self)

            # self.runButton = runButton(TouchSensor(Port))
            self.gyro = self.init(GyroSensor, Port.S1,
                                  Direction.COUNTERCLOCKWISE)
            self.Llight = self.init(LightSensor, Port.S3)
            self.Rlight = self.init(LightSensor, Port.S4)

            self.lift = Forklift(self, self.RMmotor, 11, 40, 8)
            self.drive = DriveBaseFull(self, self.Lmotor, self.Rmotor, self.gyro,
                                       56, 104, self.runButton, Llight=self.Llight,
```

```
                                    Rlight=self.Rlight)

            self.menu = {
                "runs": [["powerPlantRun", "windmillRun", "solarRun", "oilRun",
                "hydroRun", "toyRun"], [powerPlantRun(self), windmillRun(self), solarRun(
                self), oilRun(self), hydroRun(self), toyRun(self)]],
                "left": [None, None, None, None, None, None],
                "utility": [["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
                 self.drive.gyroDrift, self.drive.tyreClean]],
                "pages": ["runs", "utility"]
            }

            self.display = [self.drive.getHead]
            self.stopList = [self.drive, self.lift, self.LMmotor, self.RMmotor]

            # self.xlift = forklift(Motor(Port.B))
            # self.ylift = forklift(Motor(Port.C))
            # self.lift = doubleForklift(self.xlift, self.ylift)

        elif self.name == "apollo":
            self.SPEED_LIST_COUNT = 2000
            self.ACCELERATION = 250
            self.STARTSPEED = 50
            self.TURN_SPEED_MIN = 20
            self.TURN_SPEED_MAX = 180
            self.LIGHTCAL_CONF = "apollo.cal"

            self.Lmotor = self.init(
                Motor, Port.B, self, Direction.COUNTERCLOCKWISE)
            self.Rmotor = self.init(
                Motor, Port.A, self, Direction.COUNTERCLOCKWISE)
            self.LMmotor = self.init(Motor, Port.C, self)
            self.RMmotor = self.init(Motor, Port.D, self)

            # self.runButton = runButton(TouchSensor(Port))
            self.gyro = self.init(GyroSensor, Port.S1,
                                  Direction.COUNTERCLOCKWISE)
            self.Llight = self.init(LightSensor, Port.S3)
            self.Rlight = self.init(LightSensor, Port.S4)

            # self.lift = forklift(self, motor(self,
            #                                  Port.D, gears=[[12, 20], [28, 20], [8,
            40]]), 110)

            self.drive = DriveBaseFull(self, self.Lmotor, self.Rmotor, self.gyro,
                                       56, 104, self.runButton, Llight=self.Llight,
                                       Rlight=self.Rlight)

            self.menu = {
                "runs": [["powerPlantRun", "windmillRun", "solarRun", "oilRun",
                "hydroRun", "toyRun"], [powerPlantRun(self), windmillRun(self), solarRun(
                self), oilRun(self), hydroRun(self), toyRun(self)]],
                "left": [None, None, None, None, None, None],
                "utility": [["lightCal", "gyrodrift", "tyreClean"], [self.drive.lightCal,
                 self.drive.gyroDrift, self.drive.tyreClean]],
                "pages": ["runs", "utility"]
            }

            self.display = [self.drive.getHead]
            self.stopList = [self.drive, self.LMmotor, self.RMmotor]

            # self.xlift = forklift(Motor(Port.B))
            # self.ylift = forklift(Motor(Port.C))
            # self.lift = doubleForklift(self.xlift, self.ylift)
        else:
            self.ev3.screen.clear()
            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 
```