Логика для безопасного управления дроном:

import requests

from math import sin, cos, radians

class DroneAPI:

def \_\_init\_\_(self, api\_key):

self.api\_key = api\_key

self.base\_url = "https://droneapi.com/v1/"

# логика безопасности

self.max\_distance = 100

self.min\_battery = 20

self.current\_battery = 100

# текущая позиция дрона

self.current\_x = 0

self.current\_y = 0

self.current\_z = 0

def take\_off(self):

# проверки перед взлетом

if self.current\_battery < self.min\_battery:

print("Battery too low for takeoff!")

return

url = self.base\_url + "takeoff"

params = {"api\_key": self.api\_key}

response = requests.post(url, params=params)

if response.status\_code == 200:

self.current\_z = 10 # дрон взлетел

print("Drone took off successfully.")

else:

print("Takeoff failed!")

return response.status\_code == 200

def move(self, direction, distance):

# проверка дистанции для безопасности

if distance > self.max\_distance:

print(f"Distance {distance} exceeds max distance {self.max\_distance}")

return

# расчет новой позиции

if direction == "forward":

new\_x = self.current\_x + distance \* cos(radians(self.current\_yaw))

new\_y = self.current\_y + distance \* sin(radians(self.current\_yaw))

elif direction == "back":

# аналогично в обратную сторону

# отправка запроса на перемещение

url = self.base\_url + "move"

params = {

"api\_key": self.api\_key,

"x": new\_x,

"y": new\_y

}

response = requests.post(url, params=params)

if response.status\_code == 200:

self.current\_x = new\_x

self.current\_y = new\_y

else:

print("Move failed!")

return response.status\_code == 200

# остальные методы управления

my\_drone = DroneAPI("ABCD123")

my\_drone.take\_off()

my\_drone.move("forward", 30)

my\_drone.move("back", 20)

my\_drone.land()

Здесь мы добавили:

* Проверки перед взлетом и перемещением для безопасности
* Расчет новой позиции при перемещении
* Отслеживание текущей позиции дрона в экземпляре класса

Таким образом реализуется более сложная логика полета с учетом различных сценариев. Код становится безопаснее и менее подвержен ошибкам.

Основные преимущества:

* Проверка ограничений перед выполнением команд
* Расчет и отслеживание позиции для плавного перемещения
* Защита от случайных опасных команд управления

Такой подход позволяет добавлять всё больше логики в обертку для полноценного автономного управления дроном.