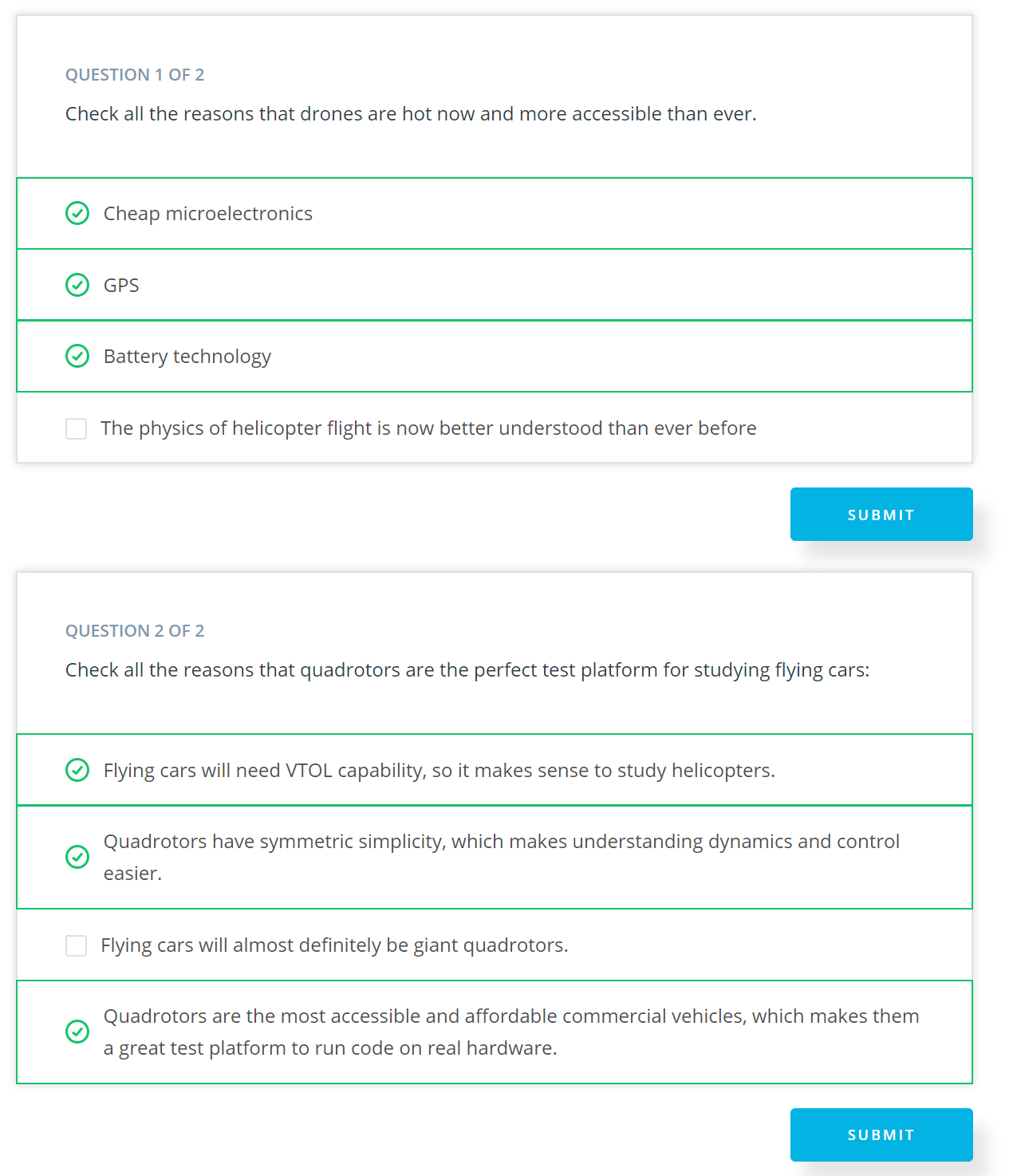
# Autonomous Flight

* Flight is inherently a dynamic process which requires active control constantly.
* Elmer Sperry invented the very first autopilot system.
* V-22 Osprey, Harrier Jump Jet, F-35B with lift fan all have VTOL capabilities.



## Quadrotor Component:

* A simple helicopter has two rotors, a main rotor and a tail rotor. If there is no tail rotor, the law of conservation of angular momentum will cause the helicopter to spin in a direction that is opposite to that of the spinning rotor.
* In a quadcopter design, rotors that sit opposite form one another spin in the same direction.
* Onboard flight computer, GPS, IMU, camera, battery etc.

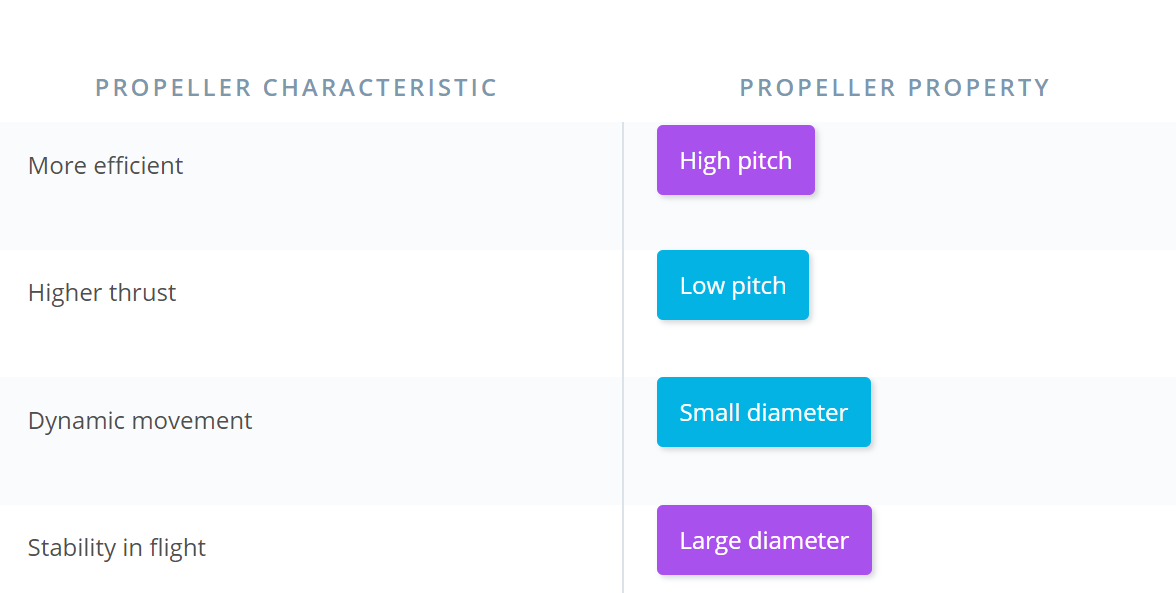
## Motor/Speed Controller

<http://www.quantumdev.com/brushless-motors-vs-brush-motors-whats-the-difference/>

<https://en.wikipedia.org/wiki/Electronic_speed_control>

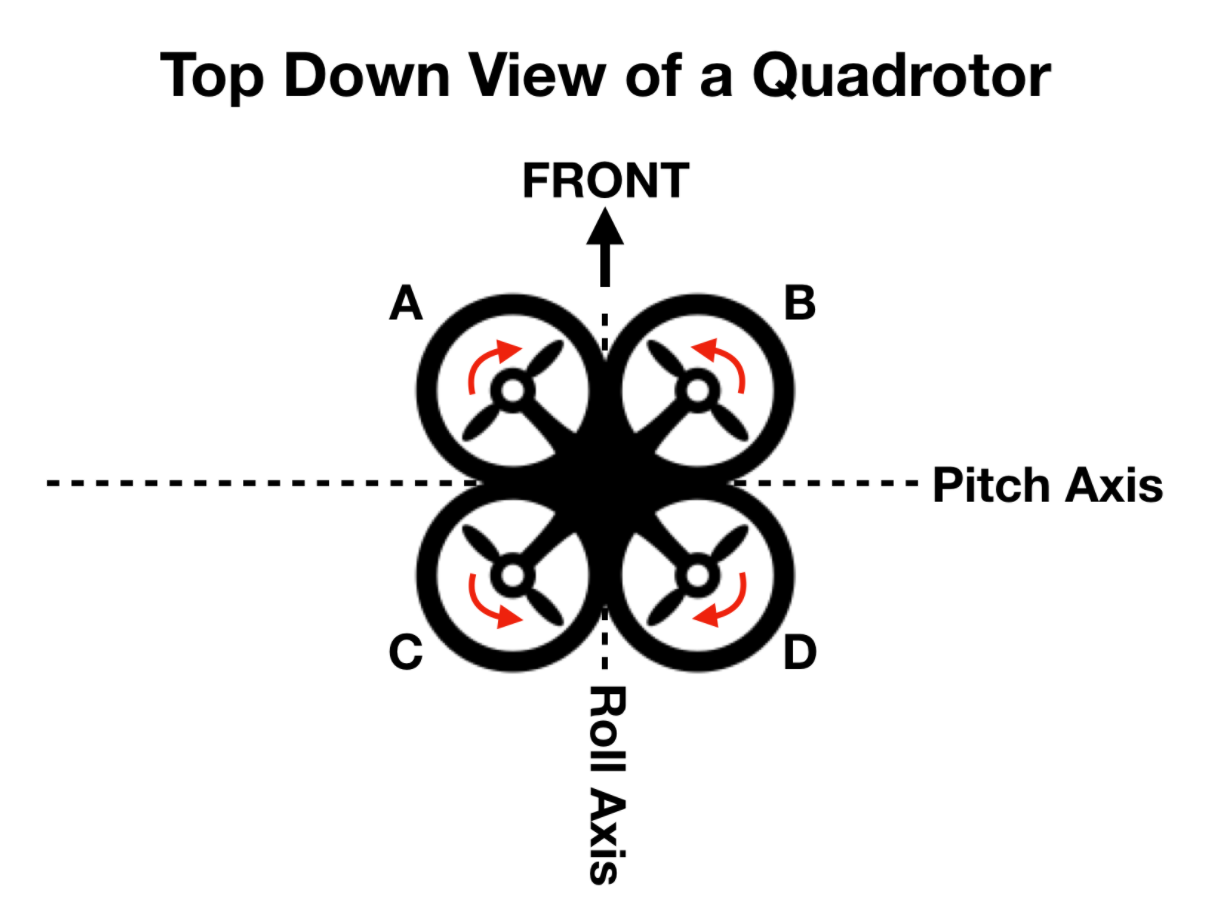
Kv = RPMs / voltage

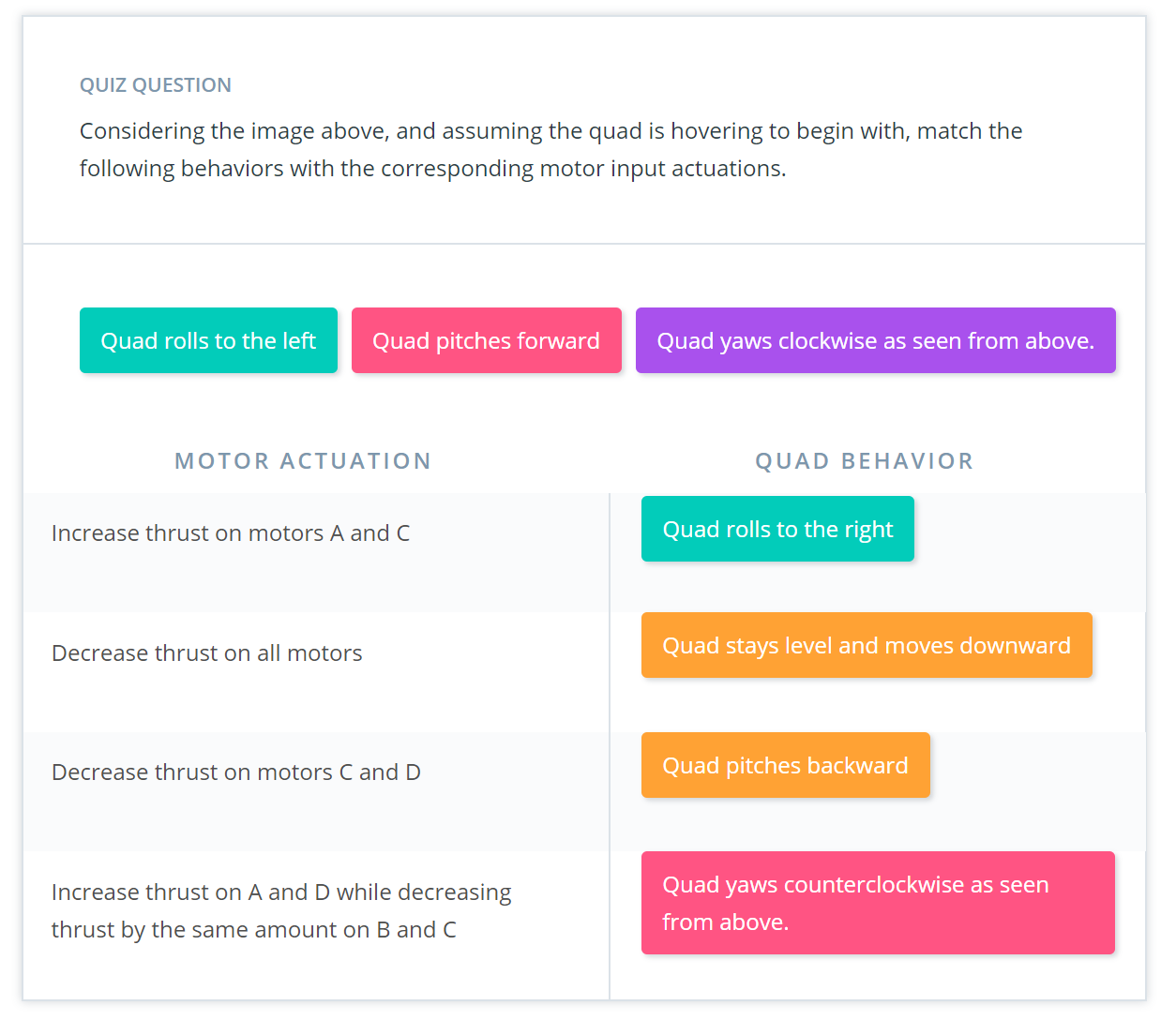
## Propeller



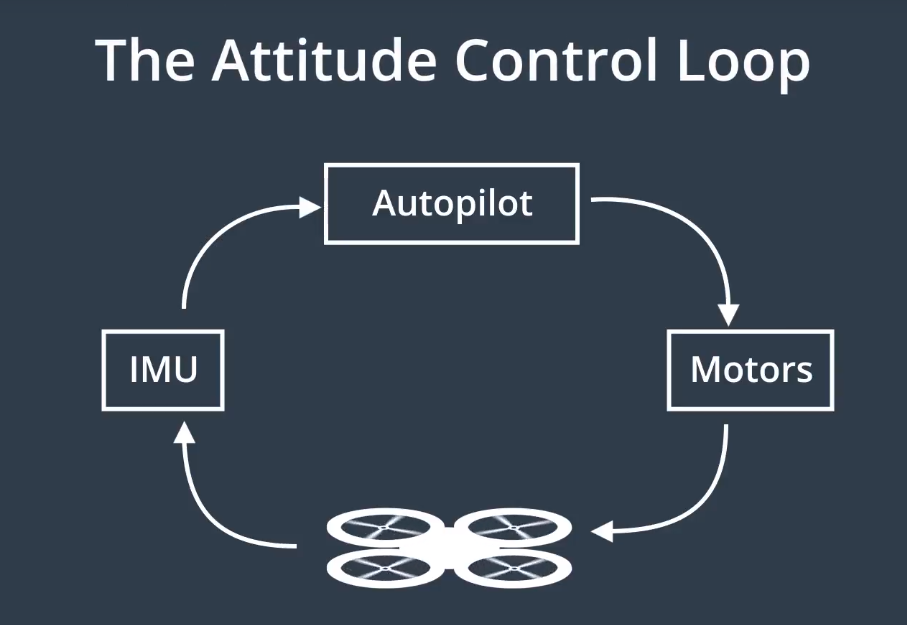
* High pitch propeller are more efficient and suitable for large motion but provides less thrust for hovering.

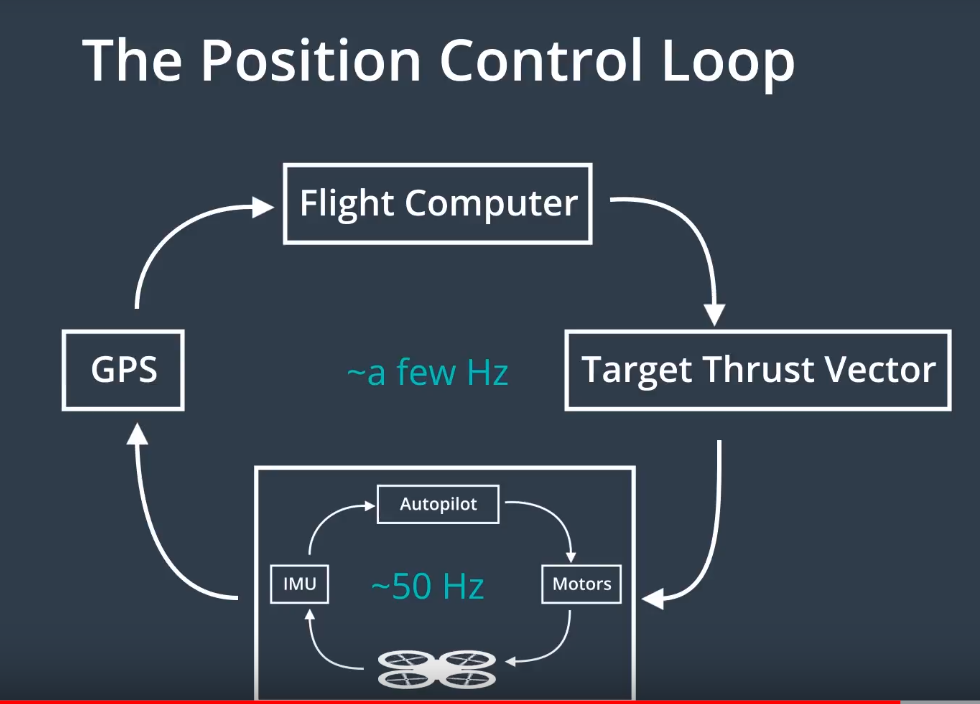
## Alttitude Control





* Attitude of a quadcopter is measured by onboard IMU(like a gyroscope)





# Project Backyard Flyer

**Lesson Overview**

In this lesson you'll learn how to control a drone with code. More specifically, you'll learn how to write *event-driven* code that controls a drone.

This event-driven programming paradigm will allow you to write drone programs that respond dynamically to a drone's changing and unpredictable world.

**Outline**

1. **Simulator Exploration** - first, you will become familiar with the drone simulator you'll be using throughout this program.
2. **Flight Computer Programming** - you'll review the roles of a drone's flight computer and autopilot so you understand how the flight computer code you'll be writing actually communicates with the drone's low-level autopilot.
3. **Event Driven Programming** - you'll explore the concepts underlying event driven programming by reading and then modifying Python code for an EventDrivenChatBot class.
4. **A Simple Flight Plan** - you'll see what event-driven programming looks like in the context of a real (but simple) flight plan.
5. **Phases of Flight** - you'll learn how we represent the "lifecycle" of a flight plan with a finite state machine.
6. **Backyard Flyer** - At the end of this lesson you'll actually implement your own flight plan within the event-driven programming paradigm you've been learning about.

## Communicating with the Drone using the API

In order to use the Drone API to communicate with the simulator. You first need to open the simulator.

Next, you will need to activate the environment via your terminal using the following command

source activate fcnd

Now the drone can be manually started from a python / ipython shell! From the terminal start ipython:

ipython

Now you can initialize the drone with the following commands:

**from** udacidrone **import** Drone

**from** udacidrone.connection **import** MavlinkConnection

conn = MavlinkConnection('tcp:127.0.0.1:5760', threaded=**True**)

drone = Drone(conn)

drone.start()

Now take control of the drone and arm the rotors.

drone.take\_control()

drone.arm()

Now set the drone's "home position"

drone.set\_home\_position(drone.global\_position[0],

drone.global\_position[1],

drone.global\_position[2])

And now you can take off (to a height of 3 meters)!

drone.takeoff(3)

Once you're in the air, you can fly around by commanding the drone to waypoints.

drone.cmd\_position(5,0,3,0)

Drone Commands

There are many commands you can issue to the drone through this API. Some of them include...

start(): Start receiving messages from the drone. If the connection is not threaded, this will block the code

stop(): Terminate the connection with the drone and close the telemetry log

take\_control(): Set the command mode of the quad to guided.

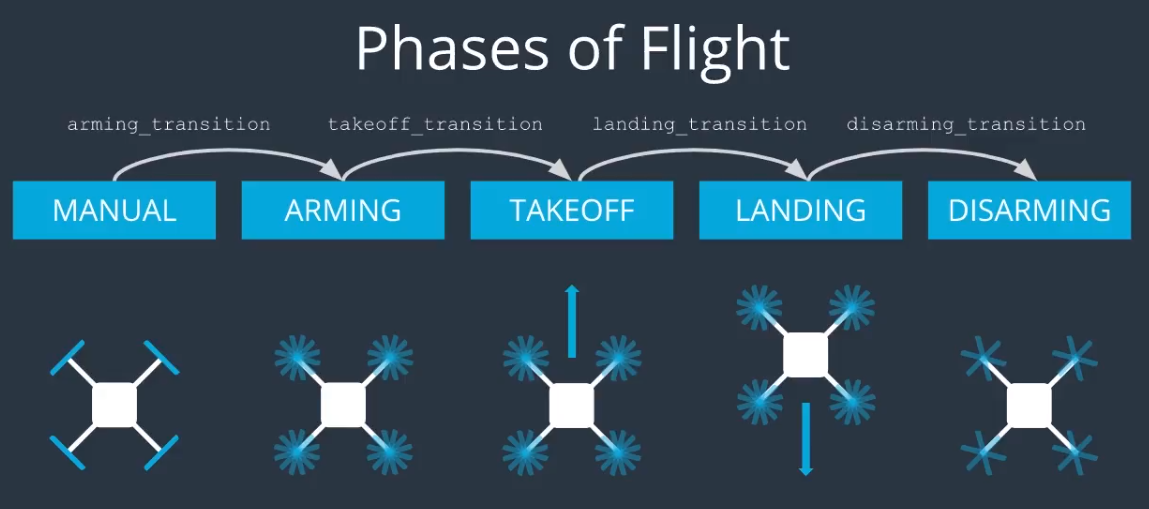
release\_control(): Set the command mode of the quad to manual.

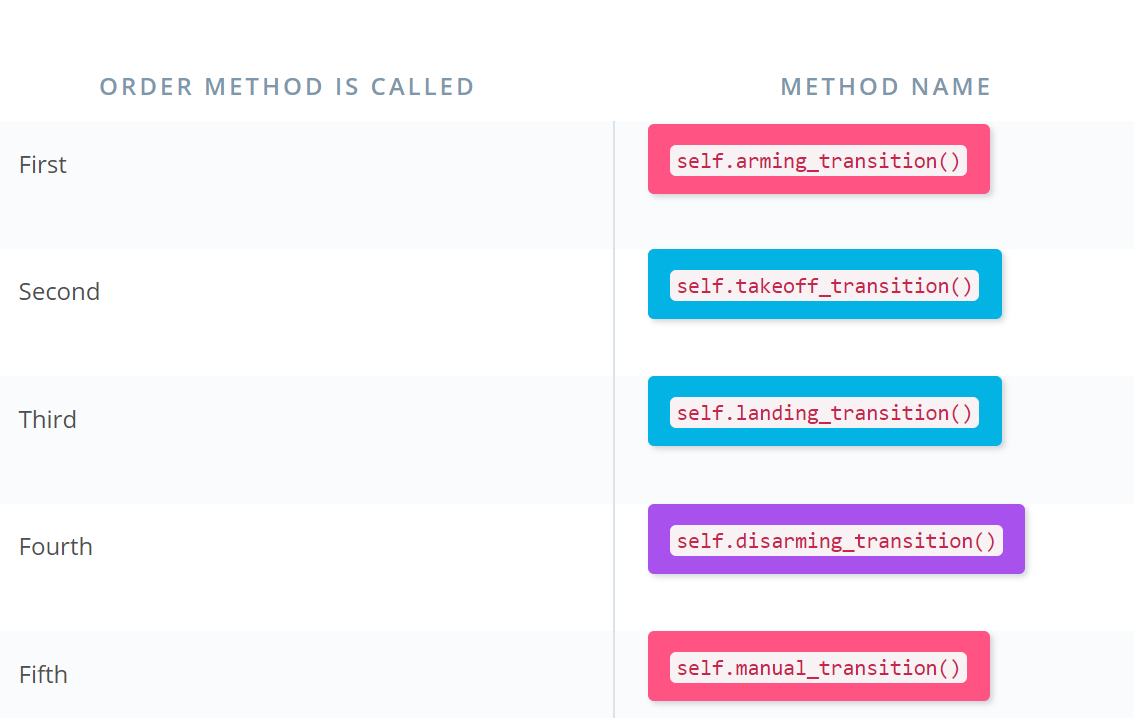
arm(): Arms the motors of the quad, the rotors begin spinning. The drone cannot takeoff until armed.

disarm(): Disarms the motors of the quad. The quadcopter cannot be disarmed in the air.

cmd\_position(north, east, down, heading): Command the vehicle to travel to the local position (north, east, down). Also commands the quad to maintain a specified heading.

## Phases of Flight





Cd /d E:\BackUp\Study\_Backup\Storage\_B\UDACITY\Course\_Flying\_Car\Term\_1\FCND-Term1-Starter-Kit\FCND-Backyard-Flyer