# HoverGames LED Tutorial - Description

This example shows how you can make the RGB LED blink in different colors with different frequency’s and slow, fast, breathe modes.

# HoverGames LED Tutorial - Challenge

* Change the color of RGB LED
* Change the blink mode
* Change the number of blinks
* Let the RGB LED blink 5 times in RED, 10 times in ORANGE and 5 times in GREEN



# HoverGames LED Tutorial - Code

1. #include <px4\_config.h>
2. #include <px4\_posix.h>
3. #include <unistd.h>
4. #include <stdio.h>
5. #include <poll.h>
6. #include <string.h>
8. #include <uORB/uORB.h>               // asynchronous messaging API
9. #include <uORB/topics/led\_control.h> // uORB topic for LED control
11. \_\_EXPORT **int** hg\_led\_main(**int** argc, **char** \*argv[]); // export main for calling from another function/thread
13. **int** hg\_led\_main(**int** argc, **char** \*argv[])
14. {
15. PX4\_INFO("Hello HoverGames LED!"); // print in console
17. **struct** led\_control\_s my\_led\_control; // structure with led\_control parameters
18. memset(&my\_led\_control, 0, **sizeof**(my\_led\_control)); // fill structure with 0
20. orb\_advert\_t led\_control\_pub = orb\_advertise(ORB\_ID(led\_control),   
     &my\_led\_control); // advertise led\_control topic
22. my\_led\_control.num\_blinks = 10;                      // blinks
23. my\_led\_control.priority = LED\_CONTROL\_MAX\_PRIORITY;  // priority
24. my\_led\_control.mode = LED\_CONTROL\_MODE\_BLINK\_NORMAL; // LED mode
25. my\_led\_control.led\_mask = 0xff;              // select LEDs - 0xff for all
26. My\_led\_control.color = LED\_CONTROL\_COLOR\_GREEN;      // color
28. orb\_publish(ORB\_ID(led\_control), led\_control\_pub, &my\_led\_control); // publish the message to uORB topic
30. PX4\_INFO("HoverGames LED exit"); // print in console
32. **return** 0; // return to calling function/thread
33. }

# HoverGames Motortest Tutorial - Description

This example shows how you can make the motors spin at a defined speed.

# HoverGames Motortest Tutorial - Challenge

* Start the motors at PWM signal rate of 1200uSec
* Determine the minimum speed of the motors
* Increase the motor speed every 5 seconds from 1000 to 1500 by steps of 100
* Make all motors spin simultaneously



# HoverGames Motortest Tutorial - Code

1. #include <px4\_config.h>
2. #include <px4\_posix.h>
3. #include <unistd.h>
4. #include <stdio.h>
5. #include <poll.h>
6. #include <string.h>
8. #include "drivers/drv\_pwm\_output.h"
10. \_\_EXPORT **int** hg\_motortest\_main(**int** argc, **char** \*argv[]); // export main for calling from another function/thread
12. **int** hg\_motortest\_main(**int** argc, **char** \*argv[])
13. {
14. PX4\_INFO("Hello HoverGames MOTORTEST"); // print in console
15. PX4\_INFO("Press CTRL-C or 'c' to abort");
17. **struct** pollfd fd\_motortest;
18. fd\_motortest.fd = 0; // Open console directly to grab CTRL-C signal
19. fd\_motortest.events = POLLIN;
21. **const** **char** \*dev = PWM\_OUTPUT0\_DEVICE\_PATH;
22. **int** fd = px4\_open(dev, 0);
24. **int** output = 1;
25. **char** c;
27. px4\_ioctl(fd, PWM\_SERVO\_SET\_MODE, PWM\_SERVO\_ENTER\_TEST\_MODE); // Start PWM
29. **while** (output) {
30. px4\_ioctl(fd, PWM\_SERVO\_SET(0), 1100); // Set motor 0 PWM to 1200
32. // Read stdin. If CTRL-C is entered, reset PWM and stop the loop
33. **if** (poll(&fd\_motortest, 1, 0) > 0) {
34. read(0, &c, 1);
36. **if** (c == 0x03 || c == 0x63 || c == 'q') {
37. px4\_ioctl(fd, PWM\_SERVO\_SET(0), 900); // Reset motor 0 PWM to 900
38. output = 0;
39. }
40. }
41. }
43. px4\_ioctl(fd, PWM\_SERVO\_SET\_MODE, PWM\_SERVO\_EXIT\_TEST\_MODE); // Exit PWM Mode
45. PX4\_INFO("HoverGames MOTORTEST exit"); // print in console
47. **return** 0; // return to calling function/thread
48. }

# HoverGames Battery Tutorial - Description

This example shows how you can read the battery voltage and current and print this information on the MavLink console.

# HoverGames Battery Tutorial – Challenge

* Read the battery voltage and current
* Read other parameters of the battery

Remember look at <uORB/topics/battery\_status.h> for the structure of the uORB message topics.

# 

# HoverGames Battery Tutorial - Code

1. #include <px4\_config.h>
2. #include <px4\_posix.h>
3. #include <unistd.h>
4. #include <stdio.h>
5. #include <poll.h>
6. #include <string.h>
8. #include <uORB/uORB.h>                  // asynchronous messaging API
9. #include <uORB/topics/battery\_status.h> // uORB topic for battery status
11. \_\_EXPORT **int** hg\_battery\_main(**int** argc, **char** \*argv[]);
13. **int** hg\_battery\_main(**int** argc, **char** \*argv[])
14. {
15. PX4\_INFO("Hello HoverGames BATTERY!"); // print in console
17. **int** battery\_sub = orb\_subscribe(ORB\_ID(battery\_status)); // subscribe to uORB
18. orb\_set\_interval(battery\_sub, 200);  // set the interval to 200 ms
20. px4\_pollfd\_struct\_t fd\_battery;
21. fd\_battery.fd = battery\_sub;
22. fd\_battery.events = POLLIN;
24. **int** counter = 20;
26. **for** (**int** i = 1; i <= counter; i++)
27. {
28. **int** poll\_ret = px4\_poll(&fd\_battery, 1, 1000); // wait for sensor update
29. **if** (poll\_ret == 0) // this means none of our providers is giving us data
30. {
31. PX4\_ERR("Got no data within a second");
32. }
33. **else**
34. {
35. **if** (fd\_battery.revents & POLLIN)
36. {
37. **struct** battery\_status\_s battery; // obtained data
38. orb\_copy(ORB\_ID(battery\_status), battery\_sub, &battery);
39. PX4\_INFO("VOLTAGE:\t %2.2f", (**double**)battery.voltage\_v);
40. PX4\_INFO("CURRENT:\t %2.2f", (**double**)battery.current\_a);
41. }
42. }
43. }
44. PX4\_INFO("HoverGames BATTERY exit"); // print in console
46. **return** 0;
47. }

# HoverGames Gyro Tutorial - Description

This example shows you how to read the gyro sensor and print the values in form of a table.

# HoverGames Gyro Tutorial - Challenge

* Change the counters



# HoverGames Gyro Tutorial - Code

1. #include <px4\_config.h>
2. #include <px4\_posix.h>
3. #include <unistd.h>
4. #include <stdio.h>
5. #include <poll.h>
6. #include <string.h>
8. #include <uORB/uORB.h>               // asynchronous messaging API
9. #include <uORB/topics/sensor\_gyro.h> // uORB topic for gyroscope sensor data
11. \_\_EXPORT **int** hg\_gyro\_main(**int** argc, **char** \*argv[]);
13. **int** hg\_gyro\_main(**int** argc, **char** \*argv[])
14. {
15. PX4\_INFO("Hello HoverGames GYRO!"); // print in console
17. **int** gyro\_sub = orb\_subscribe(ORB\_ID(sensor\_gyro)); // subscribe to uORB topic
18. orb\_set\_interval(gyro\_sub, 200);               // set the interval to 200 ms
20. px4\_pollfd\_struct\_t fd\_gyro;
21. fd\_gyro.fd = gyro\_sub;
22. fd\_gyro.events = POLLIN;
24. **int** counter = 20;
25. printf("%02i |  gyro\_x |  gyro\_y |  gyro\_z\n", counter);
26. printf("-----------------------------------\n");
28. **for** (**int** i = 1; i <= counter; i++)
29. {
30. **int** poll\_ret = px4\_poll(&fd\_gyro, 1, 1000); // wait for sensor update
32. **if** (poll\_ret == 0) // this means none of our providers is giving us data
33. {
34. PX4\_ERR("Got no data within a second");
35. }
36. **else**
37. {
38. **if** (fd\_gyro.revents & POLLIN)
39. {
40. **struct** sensor\_gyro\_s gyro; // obtained data
41. orb\_copy(ORB\_ID(sensor\_gyro), gyro\_sub, &gyro); // copy data
43. printf("%02i |  %+2.2f  |  %+2.2f  |  %+2.2f  \n", i, (**double**)gyro.x, (**double**)gyro.y, (**double**)gyro.z);
44. }
45. }
46. }
47. PX4\_INFO("HoverGames GYRO exit"); // print in console
49. **return** 0;
50. }

# HoverGames Magnetometer Tutorial - Description

This example shows you how to read the heading of the FMU and make the LED blink red when it has detected it is pointing to north.

# HoverGames Magnetometer Tutorial – Challenge

* Implement the code and test it
* Change the code so, that you can detect south
* In the code below, what is the window in degrees within which north will be detected.



# HoverGames Magnet Tutorial - Code

1. #include <px4\_config.h>
2. #include <px4\_posix.h>
3. #include <unistd.h>
4. #include <stdio.h>
5. #include <poll.h>
6. #include <string.h>
8. #include <lib/mathlib/mathlib.h>
9. #include <lib/matrix/matrix/math.hpp>
11. #include <uORB/uORB.h>                // asynchronous messaging API
12. #include <uORB/topics/vehicle\_local\_position.h>
13. #include <uORB/topics/led\_control.h>
15. **using** matrix::wrap\_2pi;
17. **extern** "C" \_\_EXPORT **int** hg\_magnet\_main(**int** argc, **char** \*argv[]);
19. **int** hg\_magnet\_main(**int** argc, **char** \*argv[])
20. {
21. PX4\_INFO("Hello HoverGames MAG!"); // print in console
23. **struct** led\_control\_s led\_control; // structure with led\_control parameters
24. memset(&led\_control, 0, **sizeof**(led\_control)); // fill the structure with 0
26. orb\_advert\_t led\_control\_pub = orb\_advertise(ORB\_ID(led\_control), &led\_control); // advertise led\_control topic
28. led\_control.num\_blinks = led\_control\_s::MAX\_PRIORITY; // blinks
29. led\_control.priority = 2;   // priority
30. led\_control.mode = led\_control\_s::MODE\_BLINK\_FAST;  // LED mode
31. led\_control.led\_mask = 0xff; // select LEDs - 0xff for all
32. led\_control.color = led\_control\_s::COLOR\_RED;       // color
34. **int** vehicle\_local\_position\_sub =orb\_subscribe(ORB\_ID(vehicle\_local\_position);
35. orb\_set\_interval(vehicle\_local\_position\_sub, 200);// set interval to 200 ms
37. px4\_pollfd\_struct\_t fd\_vehicle\_local\_position;
38. fd\_vehicle\_local\_position.fd = vehicle\_local\_position\_sub;
39. fd\_vehicle\_local\_position.events = POLLIN;

42. **int** counter = 255;
44. printf("%02i | heading \n", counter);
45. printf("---------------\n");
47. **for** (**int** i = 1; i <= counter; i++)
48. {
49. **int** poll\_ret = px4\_poll(&fd\_vehicle\_local\_position, 1, 1000);
51. **if** (poll\_ret == 0)
52. {
53. PX4\_ERR("Got no data within a second");
54. }
55. **else**
56. {
57. **if** (fd\_vehicle\_local\_position.revents & POLLIN)
58. {
59. **struct** vehicle\_local\_position\_s vehicle\_local\_position\_st;
60. orb\_copy(ORB\_ID(vehicle\_local\_position), vehicle\_local\_position\_sub, &vehicle\_local\_position\_st); // copy data into local buffer
62. int16\_t heading = (int16\_t)math::degrees  
    (wrap\_2pi(vehicle\_local\_position\_st.yaw)); // calculate orientation in degrees
64. printf("%02i |   %0.3i    | ", i, heading);
66. **if**(heading < 10 || heading > 350){
67. printf("NORTH |\n");
68. orb\_publish(ORB\_ID(led\_control), led\_control\_pub, &led\_control);
69. }**else**{
70. printf("      |\n");
71. }
72. }
73. }
74. }
76. PX4\_INFO("HoverGames MAG exit"); // print in console
78. **return** 0;
79. }