

# DJI Onboard API Linux Command Line Sample

## 说明文档

版本	时间	描述
V1.0.0	2015-05	创建

文档介绍基于 Linux 的 DJI Onboard API C++例程，编译例程后，通过 Linux 命令行终端运行程序，进行飞机的基本控制，如起飞、降落、返航等。

## 开发环境

主机平台：Ubuntu 12.04。

## 例程目录结构

例程：DJI\_Onboard\_API\_Cmdline\_Sample

目录	说明
src	源代码目录
cmake	makefile 文件目录及编译生成临时文件的目录
output	编译源代码后可执行文件的生成目录
doc	例程文档

## 主要功能函数

### ➤ 串口配置

**int Pro\_Hw\_Setup(const char \*device,int baudrate)**

函数功能：配置并打开 Linux 下串口。

函数参数：device 串口设备文件名，baudrate 串口波特率。

函数返回值：0 成功；-1 失败。

### ➤ 初始化函数

**int DJI\_Pro\_Test\_Setup(void)**

函数功能：初始化函数。包括定义 App id、key、配置通信串口等。

函数参数：无。

函数返回值：返回 0 标准成功；-1 表示失败。

### ➤ API 激活函数

#### **void DJI\_Onboard\_API\_Activation(void)**

函数功能：激活 API。

函数参数：无。

函数返回值：无。

- 获取或释放飞机控制权

#### **void DJI\_Onboard\_API\_Control(unsigned char arg)**

函数功能：激活 API 后，获取或释放控制权

函数参数：1 请求获取控制权；0 释放控制权。

函数返回值：无。

- 请求飞机起飞

#### **void DJI\_Onboard\_API\_Takeoff(void)**

函数功能：请求飞机起飞。

函数参数：无。

函数返回值：无。

- 请求飞机降落

#### **void DJI\_Onboard\_API\_Landing(void)**

函数功能：请求飞机降落。

函数参数：无。

函数返回值：无。

- 请求飞机返航

#### **void DJI\_API\_Request\_Gohome(void)**

函数功能：请求返航并降落到 Home 点。

函数参数：无。

函数返回值：无。

## 例程配置

开发者在编译使用 DJI\_Onboard\_API\_Cmdline\_Sample 前，需要根据在 DJI 网站注册获得的 app id，api level，密钥 key 以及 Linux 所使用的串口设备文件名和串口波特率按照如下图所示修改到函数 int DJI\_Pro\_Test\_Setup(void)中。

```

int DJI_Pro_Test_Setup(void)
{
    int ret;

    activation_msg.app_id = 10086;
    activation_msg.app_api_level = 2;
    activation_msg.app_ver = 1;
    memcpy(activation_msg.app_bundle_id, "1234567890123456789012", 32);
    key = "5837313ef98f1f7f1c50eebb0b06363d523a369289e042c4d00b66d8e49337a7";

    ret = Pro_Hw_Setup("/dev/ttyUSB0", 230400);
    if (ret < 0)
        return ret;
    Pro_Link_Setup();
    App_Recv_Set_Hook(App_Recv_Req_Data);
    App_Set_Table(set_handler_tab, cmd_handler_tab);
    CmdStartThread();
    Start_Simple_Task_Thread();

    return 0;
}

```

Diagram annotations:

- Red box around `activation_msg.app_id = 10086;` and `activation_msg.app_api_level = 2;` with an arrow pointing to the text "App id & App level".
- Red box around `key = "5837313ef98f1f7f1c50eebb0b06363d523a369289e042c4d00b66d8e49337a7";` with an arrow pointing to the text "Key".
- Red box around `Pro_Hw_Setup("/dev/ttyUSB0", 230400);` with an arrow pointing to the text "Uart device name & baud rate".

开发者需要注意保证程序包的波特率配置和飞机的波特率一致。

## 例程编译

在 Linux 主机下编译程序包，以下示例是在 Ubuntu 12.04 版本的主机下执行的。

打开 Linux 控制终端，输入命令 `g++ --version` 确认 Linux 安装 g++ 编译器。

```

wuyuwei@ubuntu: ~
wuyuwei@ubuntu:~$ g++ --version
g++ (Ubuntu/Linaro 4.6.3-1ubuntu5) 4.6.3
Copyright (C) 2011 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

```

进入 DJI\_Onboard\_API\_Cmdline\_Sample 的 cmake 目录，输入 **make** 命令编译程序包。

```

wuyuwei@ubuntu: ~/Desktop/DJI_Onboard_API_Cmdline_Sample/cmake
wuyuwei@ubuntu:~$ g++ --version
g++ (Ubuntu/Linaro 4.6.3-1ubuntu5) 4.6.3
Copyright (C) 2011 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

wuyuwei@ubuntu:~$
wuyuwei@ubuntu:~$ cd ~/Desktop/DJI_Onboard_API_Cmdline_Sample/
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample$
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample$ cd cmake/
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/cmake$
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/cmake$ make
g++ -Wall -O3 -Isrc/ -c ./src/main.cpp
g++ -Wall -O3 -Isrc/ -c ./src/DJI_Pro_App.cpp
g++ -Wall -O3 -Isrc/ -c ./src/DJI_Pro_Hw.cpp
g++ -Wall -O3 -Isrc/ -c ./src/DJI_Pro_Link.cpp
g++ -Wall -O3 -Isrc/ -c ./src/DJI_Pro_Test.cpp
g++ -Wall -O3 -Isrc/ -c ./src/DJI_Pro_Codec.cpp
g++ -o ../output/DJI_Onboard_API_Cmdline_Test main.o DJI_Pro_App.o DJI_Pro_Hw.o
DJI_Pro_Link.o DJI_Pro_Test.o DJI_Pro_Codec.o -lpthread
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/cmake$
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/cmake$

```

生成的 Linux 可执行文件在 DJI\_Onboard\_API\_Cmdline\_Sample/output 目录下。

## 程序运行

编译后在 DJI\_Onboard\_API\_Cmdline\_Sample/output 目录下生成可执行文件 DJI\_Onboard\_API\_Cmdline\_Test。

在 Linux 终端下使用下面命令查看程序版本：

```
./DJI_Onboard_API_Cmdline_Test -v
```

Linux 终端会输出当前程序的版本信息：

```
DJI Onboard API Cmdline Test,Ver 1.x.x
```

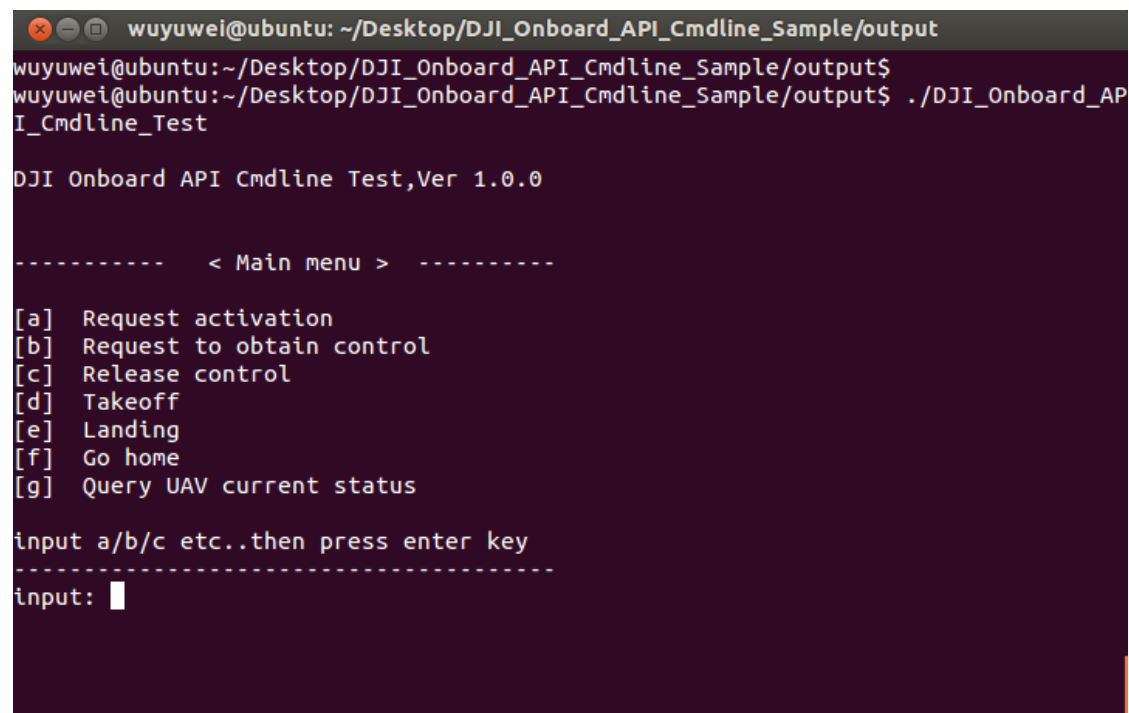
由于程序会访问 Linux 下的串口设备，开发者在运行程序前需要保证程序具有访问该串口的权限。假设串口设备名为/dev/ttyUSB0，在 Ubuntu Linux 下可以通过以下命令赋予程序访问串口的权限。

```
sudo chmod 777 /dev/ttyUSB0
```

在 Linux 终端下使用下面命令运行程序：

```
./DJI_Onboard_API_Cmdline_Test
```

程序运行后在 Linux 终端上会看到下图所示的操作菜单：



```
wuyuwei@ubuntu: ~/Desktop/DJI_Onboard_API_Cmdline_Sample/output
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/output$
wuyuwei@ubuntu:~/Desktop/DJI_Onboard_API_Cmdline_Sample/output$ ./DJI_Onboard_API_Cmdline_Test

DJI Onboard API Cmdline Test,Ver 1.0.0

----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: █
```

## 例程功能

程序运行成功后 Linux 终端会输出操作菜单，相关功能介绍如下：

```
----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: █
```

- [a]: 请求 API 激活
- [b]: 请求获得飞机控制权
- [c]: 释放飞机控制权
- [d]: 请求飞机起飞
- [e]: 请求飞机降落
- [f]: 请求飞机返航
- [g]: 查询飞机状态

## 飞行控制

进行程序飞行控制前，通过串口线连接飞机和 Linux 主机，并设置飞机在 API 控制模式下。

### ➤ 查询飞机状态

输入 g 查询飞机状态，Linux 终端如下图所示：

```
----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: g
-- Current status info: --
Activation status:[unknown]
Battery capacity:[50%]
Control device:[RC]
```

由上面信息可知道 API 的激活状态、飞机的电池剩余电量及飞机现在的受控设备。

### ➤ 激活 API

输入 a 请求激活 API，激活成功后 Linux 终端如下图所示：

```

----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: Pro_Link_Recv_Hook:Recv Session 2 ACK
Sdk_ack_cmd0_callback,sequence_number=0,session_id=2,data_len=2
[ACTIVATION] Activation result: ACTIVATION_SUCCESS
[ACTIVATION] set key DJI-DEMO AES256 KEY-lala-haha-MA

```

➤ 请求获得飞机控制权

输入 b 请求获得飞机控制权，获得成功后 Linux 终端如下图所示：

```

----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: Pro_Link_Recv_Hook:Recv Session 1 ACK
call sdk_ack_nav_open_close_callback
Recv ACK,sequence_number=13,session_id=1,data_len=2

```

获得飞机控制权后再查询飞机状态信息，可以得到如下状态提示：

```

----- < Main menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: g
-- Current status info: --
Activation status:[Activation pass]
Battery capacity:[48%]
Control device:[third party onboard device]

```

由状态信息知道，API 激活成功以及飞机的受控设备为第三方设备。

➤ 飞行控制

输入 d、e、f 控制飞机执行起飞、降落、返航等动作，命令执行成功后 Linux 终端如下图所示：

```
----- < Main Menu > -----
[a] Request activation
[b] Request to obtain control
[c] Release control
[d] Takeoff
[e] Landing
[f] Go home
[g] Query UAV current status

input a/b/c etc..then press enter key
-----
input: [DEBUG] in send
[DEBUG] send req cmd ok
Pro_Link_Rcv_Hook:Rcv Session 2 ACK
Sdk_ack_cmd0_callback,sequence_number=5,session_id=2,data_len=2
[DEBUG] CMD_RECIEVE
[DEBUG] send req status ok
Pro_Link_Rcv_Hook:Rcv Session 2 ACK
Sdk_ack_cmd0_callback,sequence_number=6,session_id=2,data_len=2
[DEBUG] rcv ack1 status ok
[DEBUG] rcv_ack 0x5
random test Cmd result: STATUS CMD EXE SUCCESS
```