Troubleshooting

Hanwha Vision Open Platform
v1.0.0
2025-01-02

Contact Information
Hanwha Vision Co., Ltd.
Hanwha Vision 6, Pangyo-ro 319beon-gil, Bundang-gu, Seongnam-si, Gyeonggi-do, 13488, KOREA
www.hanwhavision.com

Hanwha Vision America
500 Frank W. Burr Blvd. Suite 43 Teaneck, NJ 07666
hanwhavisionamerica.com

Hanwha Vision Europe
Heriot House, Heriot Road, Chertsey, Surrey, KT16 9DT, United Kingdom
hanwhavision.eu

Hanwha Vision

Table of Contents

1. Storing Files on a Flash Memory

You may read and write files under the following path:

```
/mnt/opensdk/apps/<AppName>
```

If you delete an application, the corresponding information stored in the flash memory will be deleted accordingly. If you reboot the camera without deleting the application, the stored files won't be deleted. For information to be saved even after the application is deleted, refer to "NAND Flash Space" in **6 Others** in the SDK API Document.

2. Storing Files on a SD Card

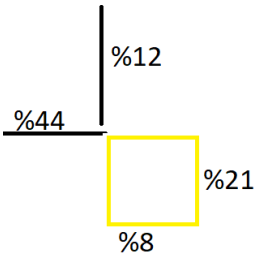
Use the following API to check the path for the mounted SD card.

```
Const INT8_N *opensdk_getSDcardStoragePath();
```

NULL will be returned if:

1. No SD card is mounted.
2. No access is available to the SD card in "IPCameraManufest.xml"

If NULL is returned despite not being applicable to the cases above, check if the SD card is activated. To activate the SD card from the camera web viewer, go to [Basic] > [Event] > [Storage], select [SD Card] from the [Storage action setup] list, and select [On] under [Record].



The space is limited to 80% of the size of the SD card. If the SD card recording is enabled, the space will be shared with a network camera's main application.

3. Implementing FTP/Email/JPEG Encoding Functions

Regarding FTP/Email/JPEG encoding, you should receive the event after calling send_event().

The behavior scenario is as below:

Open Platform Application Apphelper

send_event() ----------→ processing

recv_event() ←---------- processing result

Event type is OPENSDK_EVENT_STATUS.

TASK_STATUS.TaskName will be OPENSDK_FTP_FILE_UPLOAD for FTP, OPENSDK_JPEG_ENCODE for

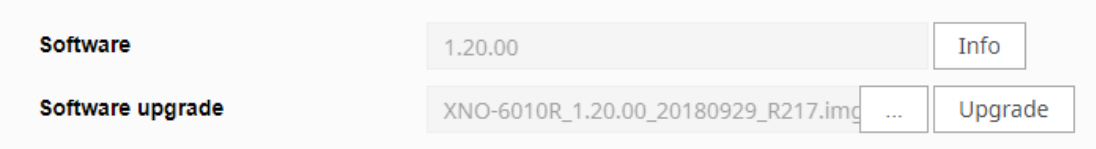
JPEG Encoding.

JPEG encoding takes about 100 ms. If you request it without waiting for the result notification, memory leaks may occur.

4. Preserving Applications after a Factory Default

During initialization to factory default, you may preserve open applications installed and their settings but reset other settings.

If you don't want to remove open applications during initialization to factory default, go to [System] > [Upgrade/Restart] on the web viewer and select [Except network parameter & Open Platform] under [Factory default].



If you click [Reset] on the camera, all settings, including network settings and open platform information, will be initialized to factory default.

5. Connecting Docker and Ubuntu by Using a Shared Folder

If you create an app by loading a Docker image on Hanwha Vision Open Platform 4.01 or higher and create a *.cap file (the app file) or source code for user OS, run the command below:

Example 1

In the example, the folder that you want to share on a current directory is 'pwd' and the shared folder is 'mnt'.

```
$> docker run --rm -it -v=`pwd`:/mnt wop:4.01
```

```
$> docker run --rm -it --volume="$PWD:/mnt" wop:4.01
```

Example 2

If you want to display debug_message, add the options below.

```
$> docker run -p 8080:8080 --rm -it --volume="$PWD:/mnt" wop:4.01
```

Example 3

In the example, VirtualBoxMountExample is the **OpenApp** folder that is currently being created.

```
root@-VirtualBox:/home/VirtualBoxShared/# docker run --rm -it
--volume="$PWD:/mnt" wop:4.01

root@0994f5fa31d7:/# cd /mnt/

root@0994f5fa31d7:/mnt# ls

VirtualBoxMountExample

root@0994f5fa31d7:/mnt# cd VirtualBoxMountExample/

root@0994f5fa31d7:/mnt/VirtualBoxMountExample# ls

IPCameraManifest.xml Makefile bin html inc res src

root@0994f5fa31d7:/mnt/VirtualBoxMountExample# touch test.ttt

root@0994f5fa31d7:/mnt/VirtualBoxMountExample# exit

exit

root@-VirtualBox:/home/VirtualBoxShared# ls

VirtualBoxMountExample

root@-VirtualBox:/home/VirtualBoxShared# cd VirtualBoxMountExample/

root@-VirtualBox:/home/VirtualBoxShared/VirtualBoxMountExample# ls
```

```
IPCameraManifest.xml Makefile bin html inc res src test.ttt
```

Reference

OpenPlatform_v4.01_Programming_Guide.pdf > 2. Developing an Application > 2.2 Installing the Open Platform SDK > Running Docker Images (Linux, Mac), Step 4.

6. Sample Application Location

When creating an app by loading a Docker image on Hanwha Vision Open Platform 4.01 or higher, we created a test application to perform verification in Hanwha Vision. This is a sample application.

The sample application is a test application used to create an app by loading a Docker image on Hanwha Vision Open Platform 4.01 and later.

The build location and build method of the sample application are as below:

Sample application location

```
$> /opt/opensdk/opensdk-4.01/SampleApplication
```

Example

```
root@20090071-L03:/opt/opensdk/opensdk-4.01/SampleApplication$ ls

test_PTZ test_Storage test_log test_sunapi_api wn7

root@20090071-L03:/opt/opensdk/opensdk-4.01/SampleApplication$ cd wn7/

root@20090071-L03:/opt/opensdk/opensdk-4.01/SampleApplication/wn7$ ls

DynamicCPU_MemoryChange PeopleCounter RawImageTest SNTest ServerPushMJPEG
test_Email test_FTP

root@20090071-L03:/opt/opensdk/opensdk-4.01/SampleApplication/wn7/$ cd
test_FTP

root@20090071-L03:/opt/opensdk/opensdk-
4.01/SampleApplication/wn7/test_FTP# make clean;make;opensdk_packager

rm -f bin/* src/buffermanager_wn7.o src/test_FTP_wn7.o
src/Opensdk_FTP_wn7.o *.cap
```

7. Connecting to a Docker Container in a Remote Server by Using Visual Studio Code in Windows 10

To connect to a Docker container on a remote server running Windows 10 in Hanwha Vision Open Platform 4.01, follow the steps below:

Prerequisites

- Windows10
- Windows Terminal (download from Microsoft Store)
- Container created by running a Docker image on a remote server

Installing WSL and Docker Desktop for windows

1. After executing Windows terminal as an administrator, run the following commands in order and reboot.

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-
Subsystem-Linux /all /norestart

dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all
/norestart
```

2. Download the Ubuntu distribution version from the Microsoft Store, install, and run it.
3. Enter **wsl –l** in the Windows terminal to confirm that Ubuntu Linux is installed.
4. To update the wsl2 Linux kernel, follow the link below:

   https://docs.microsoft.com/ko-kr/windows/wsl/install-win10#step-4---download-the-linux-kernel-update-package.

5. Enter wsl –l –v in the Windows terminal to check the current WSL version.
6. Enter wsl –set-default-version 2 to change the default to wsl2.
7. To download Docker Desktop for Windows, install, and run it, follow the link below: https://hub.docker.com/editions/community/docker-ce-desktop-windows .
8. After running, check the option to use the WSL2 based engine in the general tab of the settings page, check the Enable integration with my default WSL distro option in the resources tab, and activate Ubuntu-20.04.

Setting up an SSH-remote connection in Visual Studio Code

1. Install the Remote Development application in Visual Studio Code

2. Run the command below on the Ubuntu server where the container is running.

```
sudo apt-get update

sudo apt-get upgrade

sudo apt-get install ssh

sudo apt-get install openssh-server

sudo nano /etc/ssh/sshd_config

sudo service ssh status

sudo service ssh start

sudo ufw enable

sudo ufw allow 22

sudo ufw reload
```

3. SSH-remote connection in Visual Studio Code

```
sc config ssh-agent start=auto

net start ssh-agent
```

4. Run PowerShell on Windows and run the command below.

```
ssh-keygen -t rsa -b 4096
```

5. After entering Get-Content ..ssh\id_rsa.pub, save the output to /root/.ssh/authorized_keys on the Ubuntu server and change the permission by chmod 644 /root/.ssh/authorized_keys.
6. After pressing F1 in Visual Studio Code and clicking Remote-SSH: Connect to Host, run Configure SSH Hosts to open the config file and enter the IdentityFile information under the server to be used as below.

```
Host 192.168.38.80
```

```
HostName 192.168.38.80

User simba

IdentityFile ~/ .ssh/id_rsa
```

7. Click Remote-SSH: Connect to Host in Visual Studio Code to connect to the Ubuntu server.
8. In Visual Studio Code, press F1 and click Preferences: Open Settings (JSON) to open the settings.json file and enter the following contents at the bottom.

```
"docker.host":"ssh://simba@192.168.38.80"
```

9. Enter sudo usermod –aG docker on the Ubuntu server.

```
sudo usermod –aG docker simba
```

10. Press F1 in Visual Studio Code and click Remote-Containers: Attach to Running Container to display a list of containers that are running on the Ubuntu server. Choose the one you want and wait.
11. If a segmentation fault occurs during project build (make clean; make), refer to 9. Docker Malfunction (9p) to disable the WSL2 option.

8. Port Forwarding for Multi-SOC Cameras

Port number 8080 used by cameras with a single SOC should not be used with multiple SOCs.

Channel	Old	New	Output
CH1	http://192.168.38.188:8080/dashboard/	http://192.168.38.188:10005/	Connection OK
CH2	http://192.168.38.188:8080/dashboard/	http://192.168.38.188:10006/	Connection OK
CH3	http://192.168.38.188:8080/dashboard/	http://192.168.38.188:10007/	Connection OK
CH4	http://192.168.38.188:8080/dashboard/	http://192.168.38.188:10008/	Connection OK

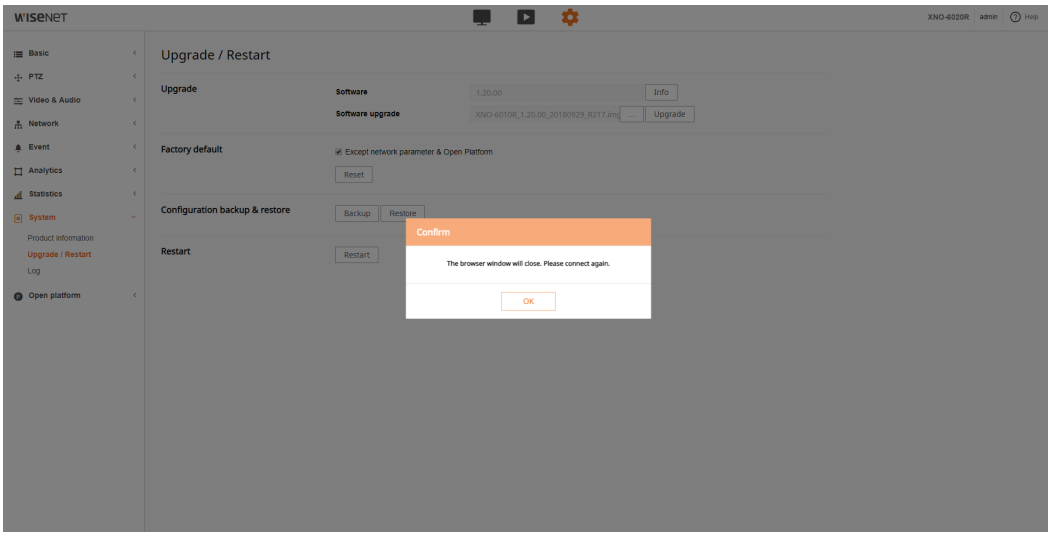
9. Docker Malfunction

When developing an application using Docker on Hanwha Vision Open Platform 4.01 or higher, seq.fault may occur if the WSL2 option is on. We recommend you disable the WSL2 option in Docker.

To disable the WSL2 option, follow the steps below:

1. Click [Settings] > [General].
2. Clear [Use the WSL 2 based engine].

3. Restart Docker.



Running error examples

```
$docker load -i wop_4.01.tar

$docker run --rm -it wop:4.01 /bin/bash

$ docker ps CONTAINER (in another terminal)

# cd /opt/opensdk/opensdk-4.01/SampleApplication/PeopleCounter/

# make

Building file: src/mjpegServer.cpp

Invoking: Cross G++ Compiler

/opt/opensdk//opensdk-4.01/armv7-wn7-linux-gnueabihf/bin/armv7-wn7-linux-
gnueabihf-g++ -O2 -g -Wall -fmessage-length=0 -c -o
'src/mjpegServer_wn7.o' 'src/mjpegServer.cpp' -I/opt/opensdk//opensdk
-4.01/armv7-wn7-linux-gnueabihf/bin/../armv7-wn7-linux
-gnueabihf/sysroot/usr/include/ -I/opt/opensdk//opensdk-4.01/common/inc/
-Iinc/

make: *** [Makefile:153: src/mjpegServer_wn7.o] Segmentation fault
```

10. Setting a Port Forwarding for Remote Debug Viewer

If Remote Debug Viewer does not work normally, check whether the Remote Debug Viewer port is open.

From Docker

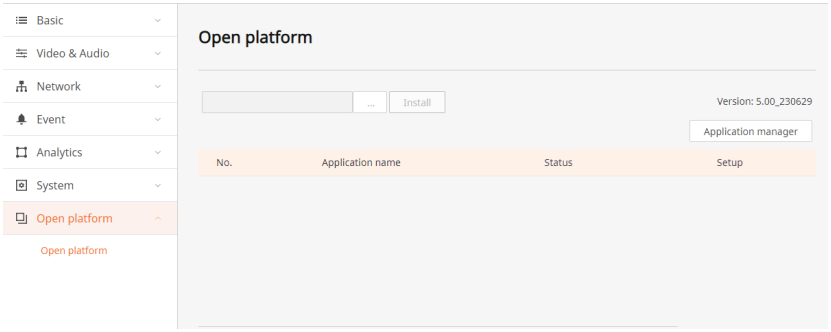
When creating a container, use the “-p” option to port forward the host port.

e.g.,) When setting the receiving port to 8080

```
# docker run -p 8080:8080 --rm -it --volume="$PWD:/mnt" wop:5.00
```

From VirtualBox

1. Go to [Settings] > [Network] > [Advanced] > [Port Forwarding].



2. On the Linux shell, run the commands below:

```
root@wisenet-
VirtualBox:/home/wisenet/Window/SampleApplication_cv2x/test_SDmount#

RemoteDebugViewer -i 192.168.38.185 -p 80 -u admin -w 5tkatjd! -d 8080 -a
test_SDmount

flag_uname 1 flag_password 1

../src/main.cpp:125 Control is here. URL is http://192.168.38.185:80/stw-
cgi/opensdk.cgi?msubmenu=debug&action=set&AppID=test_SDmount&Enable=True&
Port=8080

../src/main.cpp:135 Control is here. camera credentials are
admin:5tkatjd!
```

```
curl response: OK

../src/main.cpp:75 Control is here. Debug mode is set to ON

Data: test_SDmount one_shot

Data: test_SDmount : one_shot

Data: test_SDmount opensdk_startSDcardUse err : 2
```

3. On the test application, enter the code as follows:

e.g.,) File

```
/opt/opensdk/opensdk-
4.01/SampleApplication/test_SDmount/src/test_SDmount.cpp
```

```
void one_shot(void)
{
    debug_message("one_shot\n");
    ...

    err = OPENSDK::RECORD::opensdk_startSDcardUse(sd_number,type);
    debug_message("opensdk_startSDcardUse err : %d", static_cast<int>(
err));
    ...
```

11. The Number of Threads Running

Basically, the 3 main threads run at the same time. One is for encoded video, anther for raw video, and the other for the other events (FD, MD, VA, and so on).

Each thread sends data. If your code stays in recv_event() while processing raw video, you cannot receive raw video events anymore but can receive encoded video events.

12. Viewable Area of Encoding Video and Raw Video

Encoding video and raw video do not have the same viewable area.

For example, SNB-5004 (1.3MP camera) captures video with a 5:4 ratio, SNB-6004 (2MP camera) with

16.9, and SNB-7004 (3MP camera) with 4:3.

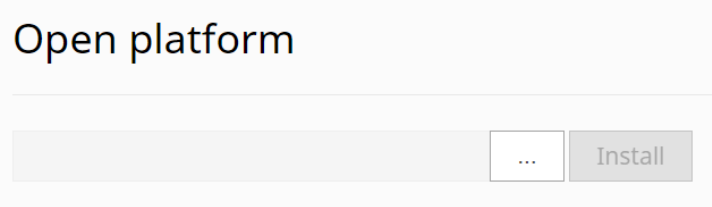
In order to create different resolutions from the original image, in the case of encoding, we just remove a part of the viewable range. For raw video, we do scaling.

13. Viewing the YUV420 Video Format

Y and UV are separated and concatenated in the memory.

The Y plane consists of serial Y data, from Y0 to Yn-1, and UV is interleaved in the plane, as U0 V0 U1 V1.

Refer to the image below (H stands for height):



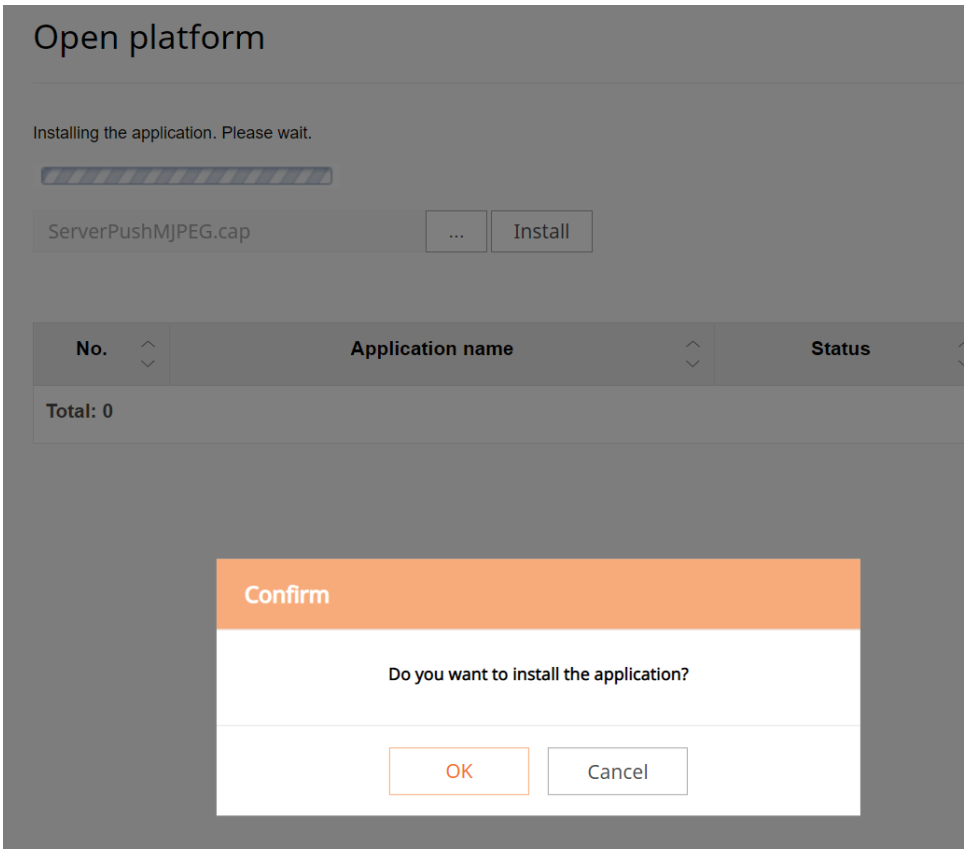
14. Logs of the Open Platform Web Interface

There is no support for web-related logs on opensdk.

The web developer of the camera development team also debugs using the logs provided by the browser.

15. Backing up the Camera Settings

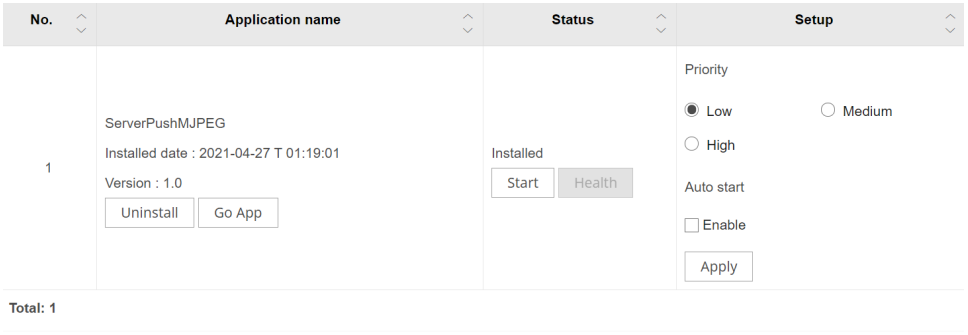
1. In the camera web viewer, click [Backup].
2. A backup file named “model name Config.bin” will be created.



16. Checking Log Files

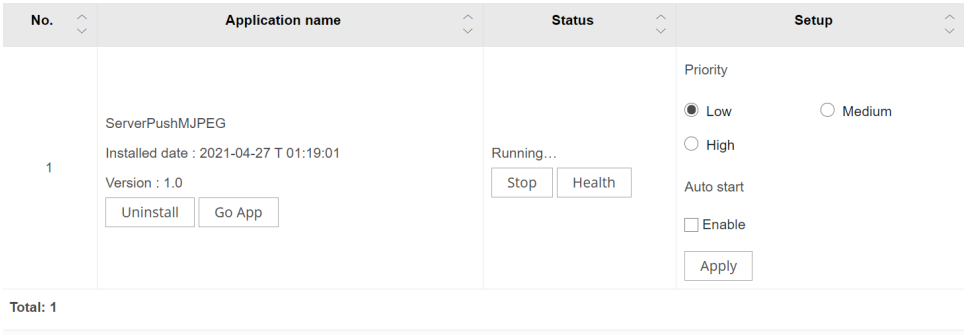
To check the system logs or events logs, follow the steps below:

1. From the [Setup] menu of the camera web viewer, select [System].
2. Click [Log].
3. Click [Backup] to save all the log data for the currently selected mode in the time stamp value created by camera in modelname-mode-camera.txt file in the download folder of the browser.



17. Checking the Camera Firmware Version

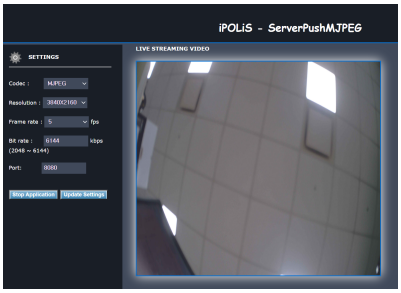
1. From the [Settings] menu of the camera web viewer, select [System].
2. Click [Upgrade / Restart].



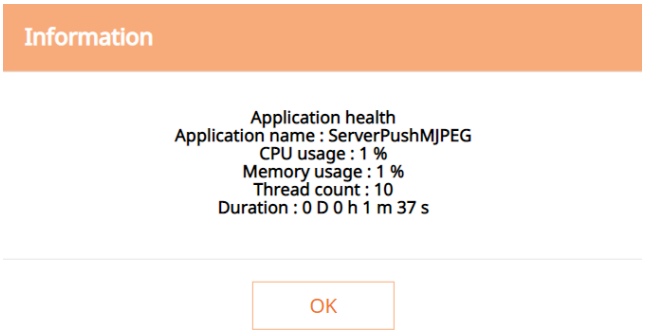
18. Adding a Library File in Open Platform

1. Build the library you want to use with the Hanwha openplatform tool chain.
   Since the space provided for the Open SDK is about 18 MB, you should build it with only the necessary parts.

2. Add the library file to any folder in your application (default: / libs)



3. Add library-specific options to the Makefile.



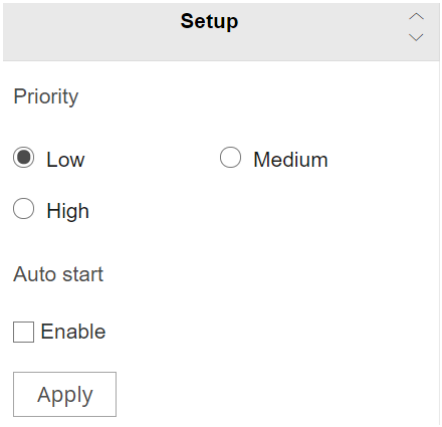
19. Testing the Dynamic Event

Dynamic Event Test is needed to a **SUNAPI Json** tool. This tool is very useful for understanding the relationship between **SUNAPI** and **OPENAPI**.

From the network camera

1. Install the **test_dynamicEvent** application in the [Open platform] webpage.

2. Click [Run] to run the application.



From the network camera

1. Start the **SunapiJson** tool and enter your ID and PW. Then click [Get Attributes]..
2. Select [eventstatus] > [check] or [monitor] or [monitordiff].

   (check: Gets the results only once.

   monitor: Connects TCP and gets the results periodically.

   monitordiff: Connects TCP and gets the results if the add schema event occurs.)
3. Choose “true” for the *Schembased* parameter.
4. Click [Send].