**Perface**

Thanks for using our product. we'll do our best to provide the best service for you. This handbook maybe contains technology mistakes or text errors. The content will be updated regularly without prior notice and the new updated part will be added in the new version.

Client SDK Instructions

**Summary**

Client SDK is a matching product of embedded hard disk video recorder(DVR), network hard disk video recorder(NVR), video server(DVS) and IPCamera, mostly used in remote device access and remote control software development.All functions below is mainly for DVR and NVR, when you integrate IPC with SDK ,just take IPC as one channel DVR device, and IPC SDK instruction page lists which functions are supported for IPC integration. The version of this product is 1.0,and its main function as the list below:

**Function Module**

Device basic info/Parameters config

Living preview

**Function Details**

device serial number,firmware version number,firmware version compiling date,core version number,hardware version number.

video formats,device ID,device name

device interface language,start password protection,screen protection,VGA parameters setting

local preview,remote preview,channel hidden,picture segmentation,picture polling

channel name,record state,alarm state(sensor alarm,network address conflict,harddisk error,disk full,system time)

electronic zoom in,picture in picture

living sound,volume adjust,mute control

Preview video/Parameterscolor adjust,adapt color to different setting time slot

Video & Audio record/Parameters settings

Capture/Parameters setting

Alarm handle/Parameters setting

character overlay:channel name,timestamp,user-defined information

area keep out

video&audio record/parameters settings

record of alarming:record time before alarming,record time after alarming,different encoding parameters setting

record switch:video switch,audio switch,video & audio binding relationship

data expiration,redundancy record,group record,whether circulation cover

record plan:record on time, record when sensor alarming,record when motion detection,record when video kept out alarming

manual record,remote control of manual record

picture measurement,picture quality,capture interval, capture numbers

manual capture,picture retrieval,picture diaplay

alarm type:sensor alarm,motion detection alarm,video kept out alarm,video lost alarm,smart analytics alarm

detection plant, detection time-lag,sensor device type,sensor device name, motion detection area & sensitivity

Alarm output/Parameters setting

alarm output:relay alarm,buzzer alarm,large picture caution,send emails,send up to center

alarm record:trigger record(specify channel),record log(alarm starts information,alarm ends information)

action with alarm:PTZ preset positions,PTZ cruise line, PTZ track

alarm from email:channel information,alarm type,attached picture(specify channel,picture numbers & time interval)

alarm from relay:switch,response schedule,response time-lag,alarm name

alarm from buzzer:switch,response schedule,response time-lag

manual alarm,remotely manual alarm

search by time:data distribution retrieval/Playback/Backupcharter, distinguish different record

search by event:event list, filter event type

search by file :file list(no partitioning record segment),lock/delete file

search by picture:picture diaplaydisplay, locked/delete pictures, picture backup(save as)

start playback:specify the start time(specify a group of channels) specify event,specify file

playback control:stop ,speed,fast backward,single frame play,reposition,exist playback,store

Network/Parameters setting

Mail functionary/Config

capture(harddisk),select area for backup

manual backup:backup by start time & stop time,backup by the specified file,montage backup,DVR & AVI formats,combination backup(multi-channels backup in one file)

automatically backup:specify time and condition,backup in external memory,backup in network service

network address setting:static address,dynamic address,PPPoE

ports-settings:HTTP port,data port,alarm port,etc.

multicast address setting

DDNS setting:customization demand,send up the period setting,etc.

parameters setting of network substream encoding:resolution,frame rate,encoded mode,picture quality,code stream limit,whether self-adaption(picture quality & fluency)

network linking setting:register user number limit,video channel number limit,whether release mainstream,black and white list

check network state,prompt network state live(normal connection,no connection,conflict),check users online,push-off users online

send emails when alarm,combine emails in a while,emails whether

PTZ control/Parameters setting

Configuration management

Disc management/Health check

System maintain

Other function

with attachment,specify to send emails, send emails manually

parameters setting of serial port ,preset position setting,cruise line setting,track setting

PTZ control:eight directions,stop,aperture,focus,zoom in,rate(128),lamplight,windshield wiper,automatically scan line

control mode:mouse 3D control,mouse control through dialog box,front panel,remote-controller,professional keyboard,remote control

protocol

local configure,remote configuration,configuration import & export,recover default configuration

format disc,delete data,specify disc group,set disc attribute(read only, read-write,redundancy,backup)

firmware upgrade,device health checked, remote upgrade,FTP upgrade

user management,permission setting

log record/retrieval/export

FTP setting

Client SDK Instructions

**IPC SDK instruction**

All supported functions for IPC:live preview, capture, sensor alarm, motion alarm, config import and export, getting or setting config, getting device time.

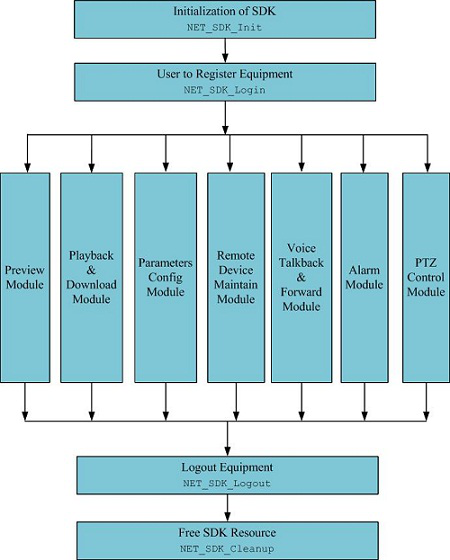
supported config for IPC:

DD\_CONFIG\_ITEM\_DEVICE\_INFO DD\_CONFIG\_ITEM\_SYSTEM\_BASIC DD\_CONFIG\_ITEM\_DATE\_TIME DD\_CONFIG\_ITEM\_DAYLIGHT\_INFO DD\_CONFIG\_ITEM\_NETWORK\_IP DD\_CONFIG\_ITEM\_NETWORK\_ADVANCE DD\_CONFIG\_ITEM\_DDNS\_SERVER\_INFO DD\_CONFIG\_ITEM\_ACCOUNT DD\_CONFIG\_ITEM\_SENSOR\_SETUP DD\_CONFIG\_ITEM\_SENSOR\_SCHEDULE DD\_CONFIG\_ITEM\_SENSOR\_ALARM\_OUT DD\_CONFIG\_ITEM\_SENSOR\_TO\_RECORD DD\_CONFIG\_ITEM\_MOTION\_SETUP DD\_CONFIG\_ITEM\_MOTION\_SCHEDULE DD\_CONFIG\_ITEM\_MOTION\_ALARM\_OUT DD\_CONFIG\_ITEM\_RELAY\_SETUP DD\_CONFIG\_ITEM\_NETWORK\_SMTP DD\_CONFIG\_ITEM\_PTZ\_PRESET DD\_CONFIG\_ITEM\_PTZ\_SETUP DD\_CONFIG\_ITEM\_VIDEO\_COLOR DD\_CONFIG\_ITEM\_ENCODE\_MASK\_MAJOR DD\_CONFIG\_ITEM\_ENCODE\_MASK\_MINOR DD\_CONFIG\_ITEM\_ENCODE\_SCHEDULE DD\_CONFIG\_ITEM\_ENCODE\_NETWORK

**Programming Guide**

This section mainly introduces functions in SDK with flow charts and text brief description.

Client SDK Instructions



**main flow of calling SDK interface**

It has seven function modules,each function module has four necessary the same flow:initialize SDK,user to login,user to logout and free SDK resource.

Initialization of SDK interface:initialize whole network SDK system,preassign internal memory etc.

Interface of setting connection timeout:this part is optional,user can set SDK network connection timeout time according to their need.If not call this interface,adopt the default in SDK.

Set callback function of receiving exception information:most module function in SDK is realized with asynchronous mode,so we provide this interface to receive exception in preview,alarming, playback and talkback module.User can call this callback function after initialization of SDK to receive and dispose exception from all module on application layer.

Get device IP address from analysis server:this interface provides the function of getting device IP address from analysis server in condition of only knowing device name and serial number. Interface of user register:realize function of

register,under successful register,returned ID is the unique identification for other functions.SDK permits maximum of register user count is 512.For device,this version permits 32 register username,meanwhile permits 128 users to register at the same time. Preview module:get real time code stream from front server,decode to display and play control function etc. Play and download module:playback or download from front server by time remotely ,later encode and store,meanwhile support resuming from break point. Parameter configuration module:set and obtain parameter of front server,including device parameter,network parameter,channel compression parameter,port parameter,alarm parameter,exception parameter,exchange information and user configuration parameter etc.

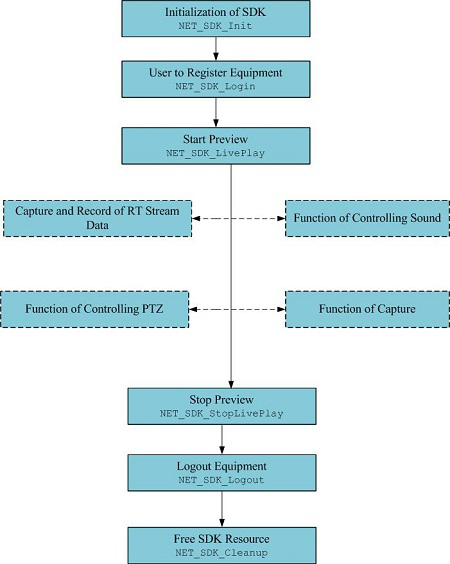
Remote device maintenance module:close device,reboot device,recover default setting, format harddisk remotely,remote upgrade,import&export of configuration file etc.

Audio talkback module:audio data talkback and obtain from front server,audio encoding method can be appointed.

Alarm module:dispose alarm signal uploaded from front server.

PTZ control module:including basic operation on PTZ,preset point,cruise and track control.SDK divides PTZ control into two methods:control by the returned handle from image preview;preview without limit,user to control PTZ through register ID.

Client SDK Instructions



**flow of preview module**

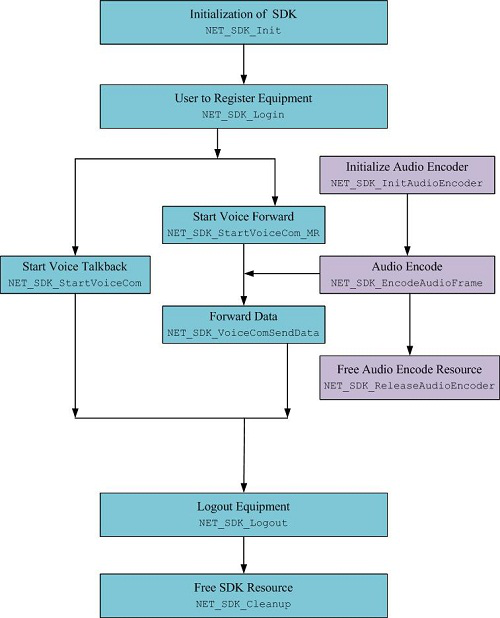
The part marked with dotted line is relative to preview module,call it in condition of open preview,the two modules are collocated and they both have their own functions.

Volume control function mainly means monopolization and volume control.

Real time data capture and record module mainly means data callback and local record for later dispose. Capture function mainly means capturing current image in decoding and store them into .BMP file.

PTZ module means PTZ control and operation function in condition of open preview,including PTZ preset point,cruise and track etc.

Client SDK Instructions



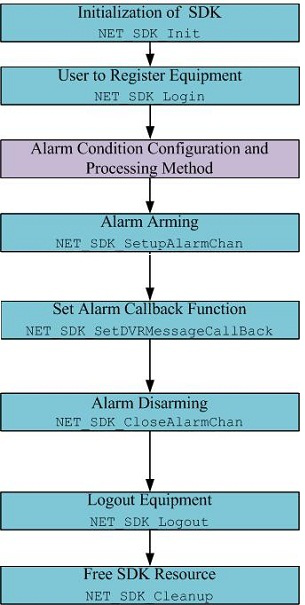
**flow of talkback and forward module**

Talkback function sends and receives audio between PC and device.Call NET\_SDK\_StartVoiceCom after device register,meanwhile user can get data which is sent by

device or collected by PC through setting callback function.

Firstly call NET\_SDK\_StartVoiceCom\_MR to start talkback(connection with device has been done and wait for sending data). Second,prepare which data to send(need to encode),flow of encoding is the part in purple,if data has been compressed in appointed method omit the encoding part.Data resource can be from PC sound card or read from file,but must be compressed with our compression algorithm. After encoding operation we can get encoded data in fixed size,then call NET\_SDK\_VoiceComSendData to send the data to device. After sending all data,call NET\_SDK\_StopVoiceCom to stop forward connection with device.

Client SDK Instructions



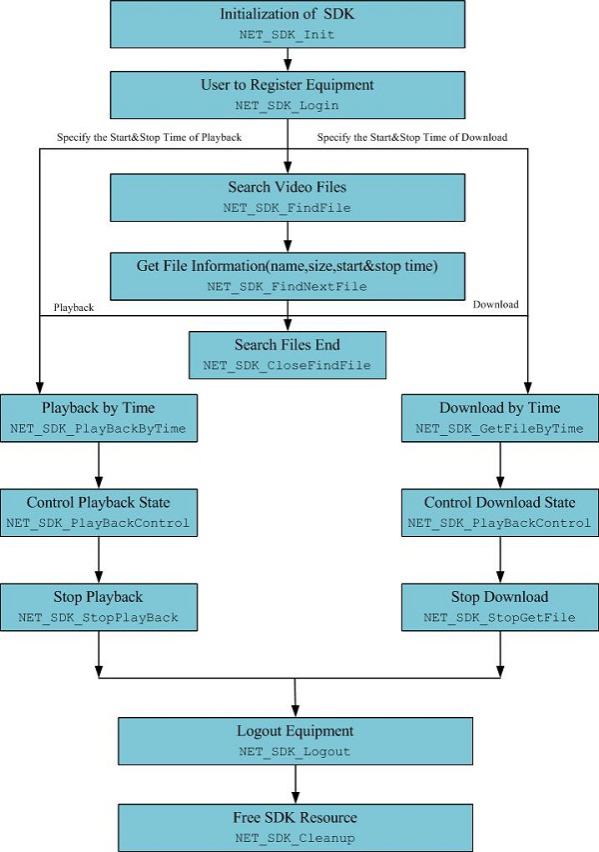
**flow of alarm module**

Alarm method in this version is arming:SDK connects device actively, sends command of uploading alarm,then device alarms and sends to SDK. From the flow chart we can know that arming method needs user

register first.The part in purple is a necessary conditon for realizing alarm information upload, mainly finishes relative alarm condition and process configuration,the interface of parameter configuration is NET\_SDK\_GetDVRConfig and NET\_SDK\_SetDVRConfig.Configurated struct for single quantity alarm is NET\_SDK\_AlarmInfo,if these parameters configuration has been done, next step is setting alarm callback function NET\_SDK\_SetDVRMessageCallBack,after above steps,set arming alarm NET\_SDK\_SetupAlarmChan. Cancel arming interface should be called after the whole alarm uploading progress.

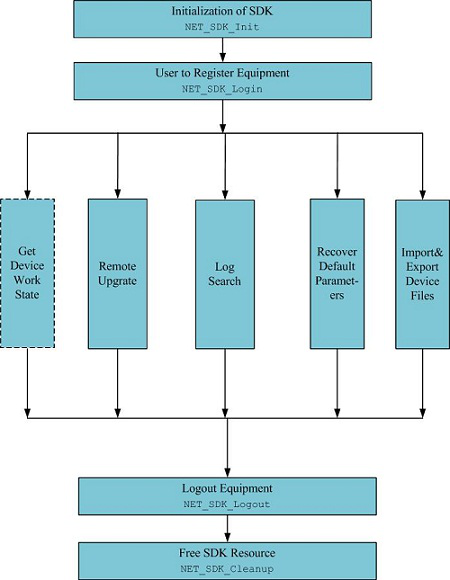
Client SDK Instructions

**flow of playback and download by time module**



When playback and downloading by time,user needn't call relative find file interface but just appoint the start time and stop time in interface. The start play command of control interface should be called after calling playback and downloading interface,then the nearest video will playback or download by appinted time range. Also user can call relative interface of finding record file to get the start time and stop time,and appoint the time parameter according to the returned time range,at last the start play command of control interface should be called too.

Client SDK Instructions



**flow of remote device maintainance module**

Remote device maintenance module includes getting device work status,remote upgrade,log search,recover device default parameter and import&export configuration file etc.

Get device work status:get device current harddisk state,channel state,alarm import and export state,local display state and audio channel state,the part marked with dotted line is reserved temporarily.

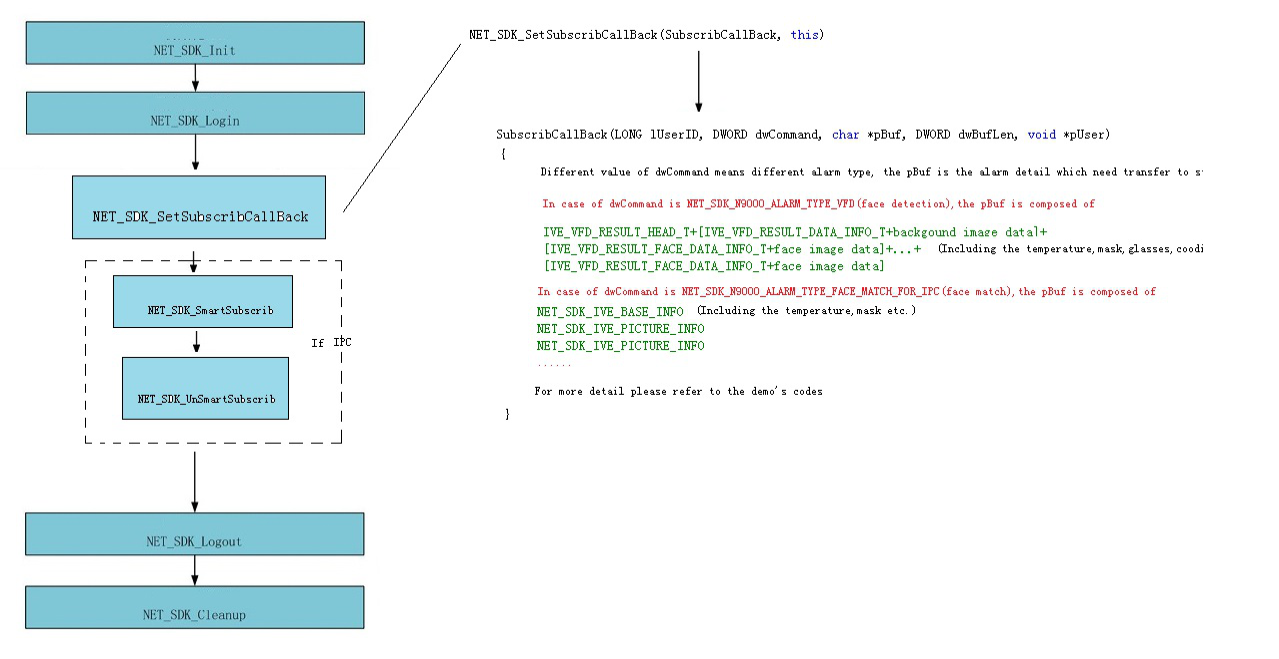
Remote upgrade:upgrade device and get the upgrade progress and state.

Search log:search current device log information,including alarm,exception,operation and log with S.M.A.R.T information.

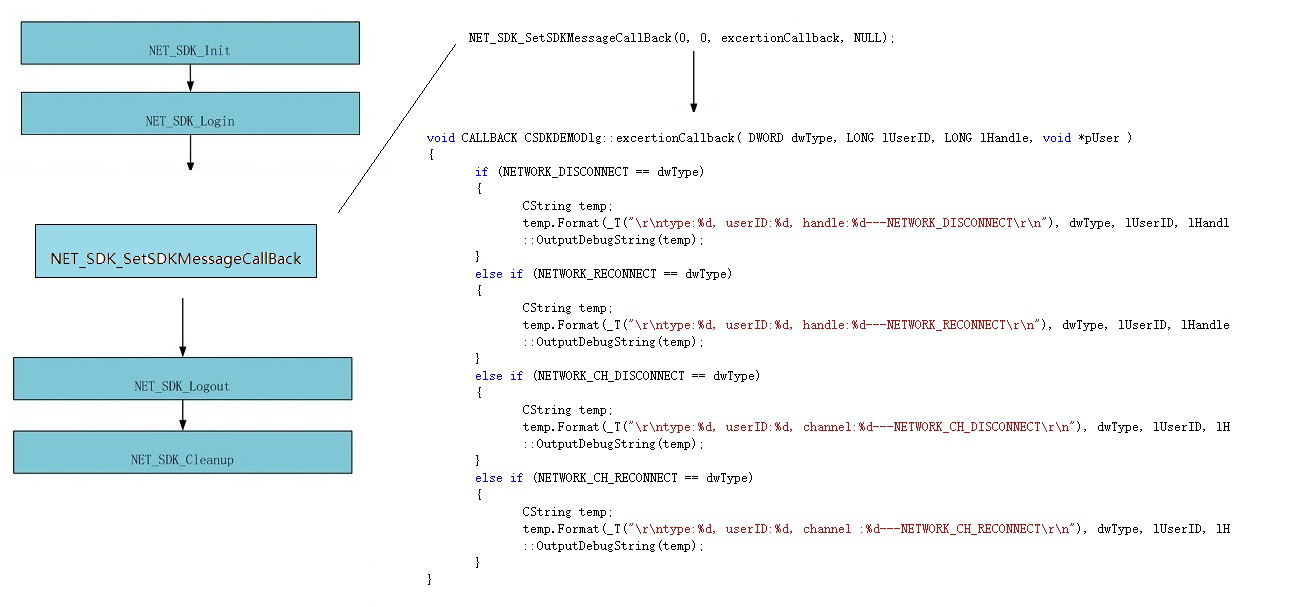
Recover default device parameter:call NET\_SDK\_RestoreConfig to recover all default parameter setting.

Import and export configuration file :export current all configuration information and store them or import the appointed configuration information.

Client SDK Instructions

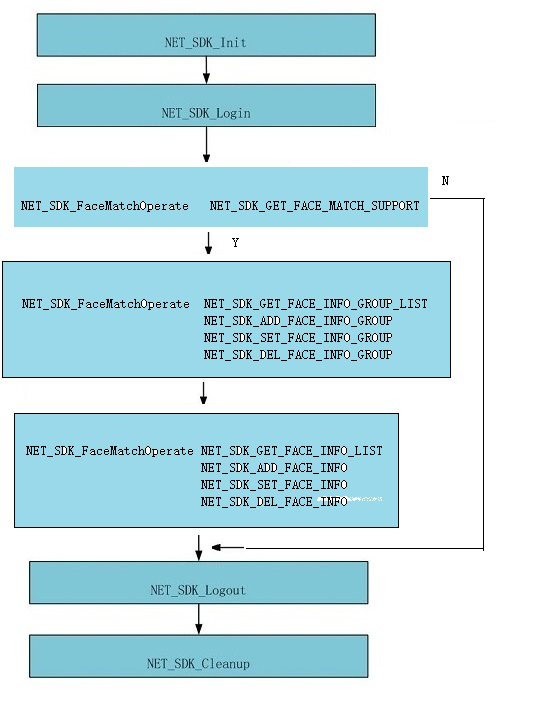
**flow of intelligent alarm**

Client SDK Instructions

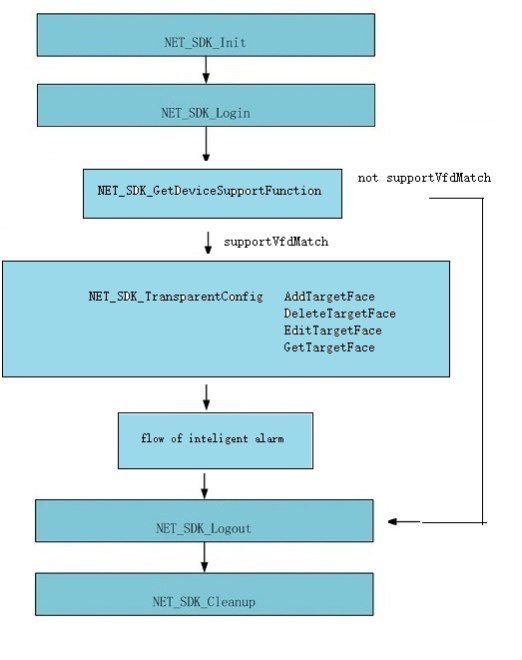
**flow of exception**

Client SDK Instructions

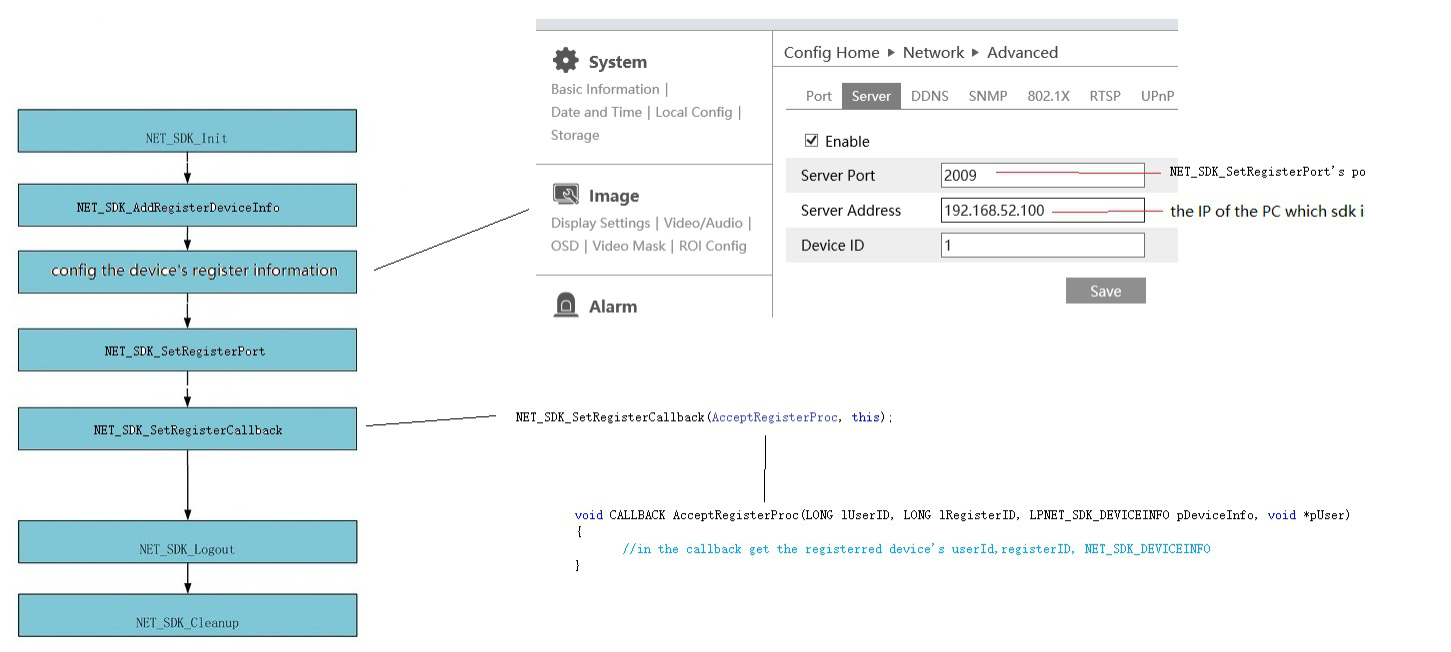
**flow of config N9000's face album target**



Client SDK Instructions

**flow of config IPC's face album target**

Client SDK Instructions

**flow of accept device's register**

**Interface Definition**

This section mainly introduces all the interface definitions involved in SDK , one single page corresponding to one interface definition,

and has a brief description for parameters and return values.

Further more the structures' definitions that involved in interface definitions are added,also one single page corresponding to one structure definition.

Client SDK Instructions

**Macro Definition**

**value of**

**macro definition** **macro** **meaning definition**

DD\_MAX\_CAMERA\_NUM 128

DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN 16

DD\_MAX\_SERIAL\_NUMBER\_LEN 64

DD\_MAX\_VERSION\_BUF\_LEN 64

DD\_MAX\_NAME\_LEN 64

DD\_MAX\_NAME\_BUF\_LEN 132

DD\_MAX\_CAMERA\_NAME\_LEN 64

the maximum number of inserted camera shooting device

the maximum byte length of inserted camera shooting device

length of serial number

length of version buffer

length of user name

buffer length of user name

the maximum

DD\_MAX\_CAMERA\_NAME\_BUF\_LEN 132

DD\_MAX\_URL\_LEN 256

DD\_MAX\_URL\_BUF\_LEN 260

DD\_MAX\_COLOR\_CFG\_NUM 3

DD\_MAX\_TEXT\_LEN 64

DD\_MAX\_TEXT\_BUF\_LEN 132

name length of inserted camera shooting device

the maximum name buffer length of inserted camera device

the maximum length of input URL

the maximum buffer length of input URL

number of stream for controlling color

the maximum length of input text

the maximum buffer length of input text

DD\_MAX\_VIDEO\_COVER\_NUM 3

DD\_MAX\_USER\_NAME\_LEN 64

DD\_MAX\_USER\_NAME\_BUF\_LEN 132

DD\_MAX\_PASSWORD\_LEN 128

DD\_MAX\_PASSWORD\_BUF\_LEN 132

DD\_MAX\_PPPOE\_ACCOUNT\_LEN 128

DD\_MAX\_PPPOE\_ACCOUNT\_BUF\_LEN 132

DD\_MAX\_DDNS\_ACCOUNT\_LEN 128

the maximum number of video override

the maximum length of user name

buffer length of user name

the maximum length of password

the maximum buffer length of password

the maximum length of PPPOE dialling number

buffer length of PPPOE dialling number

the maximum length of

DDNS number

the maximum

DD\_MAX\_DDNS\_ACCOUNT\_BUF\_LEN 132 length of DDNS number

DD\_MAX\_EMAIL\_RECEIVE\_ADDR\_NUM3

DD\_MAX\_MOTION\_AREA\_WIDTH\_NUM1920/16

number of address for receiving emails

value of the width of motion area

value of DD\_MAX\_MOTION\_AREA\_HIGHT\_NUM ((1080/16)the height

area

DD\_MAX\_PRESET\_NUM

DD\_MAX\_CRUISE\_NUM

DD\_MAX\_TRACK\_NUM

DD\_MAX\_ACCOUNT\_NUM

DD\_MAX\_BUF\_SIZE

128

32

1

64

512\*1024

number of PTZ preset points

number of PTZ cruise

number of PTZ track

the maximum number of user

size of buffer

Client SDK Instructions

**NET\_SDK\_Init**

initialize SDK,before calling other functions in SDK

BOOL NET\_SDK\_Init( );

**Return Values**

TRUE means success; FALSE means failure.To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Cleanup](#_page_78_0)

Client SDK Instructions

**NET\_SDK\_SetConnectTime**

set network connection timeout time and connection times

BOOL NET\_SDK\_SetConnectTime( DWORD *dwWaitTime*

DWORD *dwTryTimes* );

**Parameters**

*dwWaitTime*

[in] timeout time,in millisecond,its value is greater than 300. actual maximum timeout value is **connect** timeout value(**connect** timeout value depends on different system)the excess part is invalid,default value is 5 seconds.

*dwTryTimes*

[in] reconnection times(kept),default value is 3

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

SDK default timeout value is 5 seconds.

Client SDK Instructions

**NET\_SDK\_SetReconnect**

set reconnection function

BOOL NET\_SDK\_SetReconnect( DWORD *dwInterval*, BOOL *bEnableRecon*

);

**Parameters**

*dwInterval*

[in] reconnection interval,in millisecond,default value is 30 seconds

*bEnableRecon*

[in] whether to reconnect,0- no reconnect,1-reconnect,default value is 1

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface can control preview, transparent channel and reconnection functions in arming at the same time.When it isn't called SDK enables the three functions acquiescently, and the reconnection time interval is 5 seconds.

Client SDK Instructions

**NET\_SDK\_Cleanup**

before finish, the last step is to free SDK resource.

BOOL NET\_SDK\_Cleanup( );

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Init](#_page_72_0)

Client SDK Instructions

**NET\_SDK\_DiscoverDevice**

discover device automatically on LAN

long NET\_SDK\_DiscoverDevice( [NET\_SDK\_DEVICE\_DISCOVERY\_INFO](#_page_678_0)

*\*pDeviceInfo*, long *bufNum*;

long *waitSeconds*; );

**Parameters**

*\*pDeviceInfo*

[in] an array witch is needed to asign values,its size is **bufNum** ,if descovered device num is more than,the returned size is just **bufNUm**

*bufNum*

[in] size of the array *waitSeconds*

[in] time to discover devices, unit is second,this interface will be returned after **waitSeconds**

**Return Values**

Returned value is the number of discovered devices, if no device is found or discovering device gets error,the value is 0. To get error information, please refer to [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_GetDeviceInfo](#_page_467_0)

Client SDK Instructions

**NET\_SDK\_DiscoverDeviceStart**

start discover device on LAN(asynchronous, only for windows)

unsigned int NET\_SDK\_DiscoverDeviceStart( [IPTool\_SearchDataCallBack](#_page_87_0)

*SearchCallBack*, [IPTool\_SearchDataCallBackEx](#_page_89_0)

*SearchCallBackEx*, void *\*pParam*,

unsigned int *SearchTypeMask*, int nMaxRecordCount

);

**Parameters**

*SearchCallBack*

search result callback during searching, result format is xml

*SearchCallBackEx*

[in] search result callback during searching, result format is [SEARCHED\_DEVICE\_INFO](#_page_790_0) struct

*\*pParam*

pointer to user data *SearchTypeMask*

search type，refer to the table：

**type** **value**

\_SEARCH\_STANDARD 0x001

\_SEARCH\_ONVIF 0x002

\_SEARCH\_UPNP 0x004

**meanning**

standard device

onvif device

upnp device

\_SEARCH\_AIPSTAR 0x008

\_SEARCH\_DAHUA 0x010

\_SEARCH\_HIK 0x020

\_SEARCH\_UNIVIEW 0x040

\_SEARCH\_YCX 0x080

\_SEARCH\_SPECO 0x100

\_SEARCH\_ALL 0xffff

AIPSTAR device

DAHUA device

HIK device

UNIVIEW device

YCX device

SPECO device

all type

*nMaxRecordCount*

max number of searched devices

**Return Values**

Returned value is the handle of searching. To get error information, please refer to [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_DiscoverDeviceStop**

stop asynchronous searching（only for windows）

void NET\_SDK\_DiscoverDeviceStop( unsigned int *hSearch*,

);

**Parameters**

*hSearch*

the handle of searching, the return value of [NET\_SDK\_DiscoverDeviceStart](#_page_82_0)

Client SDK Instructions

**IPTool\_SearchDataCallBack**

call back of asynchronous searching

void \*IPTool\_SearchDataCallBack( char\* *hwaddr* ,

char\* *szDevIP* , int *opt* ,

const char\* *szXmlData*, void \* *pParam*,

const char \* *szRecvFromNIC* );

**Parameters**

*hwaddr*

hard ware address *szDevIP*

IP *opt*

reserve *szXmlData*

the searched device's information with xml format *pParam*

reserve *szRecvFromNIC*

reserve

**Return Values**

No return value. To get error information, please refer to [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**IPTool\_SearchDataCallBackEx**

call back of asynchronous searching

void \*IPTool\_SearchDataCallBackEx( char\* *hwaddr* ,

char\* *szDevIP* , int *opt* ,

const [SEARCHED\_DEVICE\_INFO](#_page_790_0) \* *pData* , void \* *pParam*,

const char \* *szRecvFromNIC* );

**Parameters**

*hwaddr*

hard ware address *szDevIP*

IP *opt*

reserve *pData*

the searched device's information *pParam*

reserve *szRecvFromNIC*

reserve

**Return Values**

No return value. To get error information, please refer to [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SetRegisterCallback**

callback function when device receives DVR registered local port

BOOL NET\_SDK\_SetRegisterCallback( [ACCEPT\_REGISTER\_CALLBACK](#_page_99_0) *fRegisterCBFun*

void *\*pUser* );

**Parameters**

*fRegisterCBFun*

[in] callback information when device receives DVR registered local port

*\*pUser*

[in] custom parameter passed by user

**Return Values**

TRUE means sccess; FALSE means failure. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_SetRegisterPort](#_page_95_0) [ACCEPT\_REGISTER\_CALLBAC](#_page_99_0)

[K](#_page_99_0)

Client SDK Instructions

**NET\_SDK\_SetUnRegisterCallback**

the function is a callback for the unregistered device,calls NET\_SDK\_AddRegisterDeviceInfo to notify SDK after receiving callback

BOOL NET\_SDK\_SetUnRegisterCallback(

[ACCEPT\_UNREGISTER\_CALLBACK](#_page_99_0)

*fUnRegisterCBFun*

void *\*pUser* );

**Parameters**

*fUnRegisterCBFun*

[in] callback information when device receives DVR registered local port

*\*pUser*

[in] custom parameter passed by user

**Return Values**

TRUE means sccess; FALSE means failure. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/interface/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_SetRegisterPort](#_page_95_0) [ACCEPT\_REGISTER\_CALLB](#_page_99_0)

[ACK](#_page_99_0)

Client SDK Instructions

**NET\_SDK\_SetRegisterPort**

local port when device receives DVR's register

BOOL NET\_SDK\_SetRegisterPort( WORD *wRegisterPort*

);

**Parameters**

*wRegisterPort*

[in] local port when device receives DVR's register

**Return Values**

TRUE means sccess; FALSE means failure. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_SetRegisterCallback](#_page_91_0) [ACCEPT\_REGISTER\_CALL](#_page_99_0)

[BACK](#_page_99_0)

Client SDK Instructions

**NET\_SDK\_AddRegisterDeviceInfo**

add the informations of the devices which need to auto register to sdk

BOOL NET\_SDK\_AddRegisterDeviceInfo( [REG\_LOGIN\_INFO](#_page_788_0) *\* pLoginInfo, unsigned int deviceNum*

*);*

***Parameters***

*REG\_LOGIN\_INFO*

*[in] the pointer of the devices' information deviceNum*

*[in] number of the devices*

***Return Values***

*TRUE means sccess; FALSE means failure. To get error code, please call* [*NET\_SDK\_GetLastError*](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

***See Also***

[*NET\_SDK\_SetRegisterCallback*](#_page_91_0) [*ACCEPT\_REGISTER\_CALL*](#_page_99_0)

[*BACK*](#_page_99_0)

Client SDK Instructions

**ACCEPT\_REGISTER\_CALLBACK**

callback information when device receives DVR's register

void ACCEPT\_REGISTER\_CALLBACK( LONG *lUserID*,

LONG *lRegisterID*, [LPNET\_SDK\_DEVICEINFO](#_page_681_0) *pDeviceInfo*, void *\*pUser*

);

**Parameters**

*lUserID*

[in] connection ID,other interface accesses device through this ID

*lRegisterID*

[in] device received DVR initiative register ID *pDeviceInfo*

[in] information of initiative register device *\*pUser*

[in] custom parameter passed by user

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_SetRegisterCallback](#_page_91_0) [NET\_SDK\_SetRegisterPort](#_page_95_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceIPByName**

Get the device Ip by the device name

BOOL NET\_SDK\_GetDeviceIPByName( char *\*sSerIP*,

DWORD *wSerPort*; char *\*sDvrName*; char *\*sDvrIP*

);

**Parameters**

*\*sSerIP*

[in] the IP address of IPSever *wSerPort*

[in] the port of IPSever *\*sDvrName*

[in] the device name automatically reported to IP Server

*\*sDvrIP*

[in] the device port automatically reported to IP Server

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_DiscoverDevice](#_page_80_0)

Client SDK Instructions

**NET\_SDK\_SetSDKMessageCallBack**

callback function of SDK operation exception

BOOL NET\_SDK\_SetSDKMessageCallBack( UINT *nMessage*,

HWND *hWnd*, [EXCEPTION\_CALLBACK](#_page_107_0) *fExceptionCallBack*, void *\*pUser*

);

**Parameters**

*nMessage*

[in] message *hWnd*

[in] window handle of receiving exception message *fExceptionCallBack*

[in] callback function of receiving exception message,callback current relevant information of exception

*\*pUser*

[in] user data

**Callback Function**

void (CALLBACK fExceptionCallBack)( DWORD *dwType*,

LONG *lUserID*, LONG *lHandle*, void *\*pUser* );

**Callback Function Parameters**

*dwType*

message type of exception or reconnection *lUserID*

login ID *lHandle*

relevant type handle of exception *pUser*

user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**EXCEPTION\_CALLBACK**

data callback during SDK exception

void \*EXCEPTION\_CALLBACK( DWORD *dwType*, LONG *lUserID*, LONG *lHandle*, void *\*pUser*

);

**Parameters**

*dwType*

[in] type of exception or reconnection message,refer to NET\_SDK\_EXCEPTION\_TYPE:

**Type** **Value Description**

NETWORK\_DISCONNECT 0

NETWORK\_RECONNECT 1

Disconnection

Reconnection

NETWORK\_CH\_DISCONNECT2

Channel Disconnection

NETWORK\_CH\_RECONNECT 3

Channel Reconnection

*lUserID*

[in] login ID *lHandle*

[in] corresponding type handle when exception *\* pUser*

[in] pointer to user data

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_SetVideoEffect**

Set the display parameter of the video.

BOOL NET\_SDK\_SetVideoEffect( LONG *lUserID*,

LONG *lChannel*, DWORD *dwBrightValue*,

DWORD *dwContrastValue*, DWORD *dwSaturationValue*, DWORD *dwHueValue*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] return value of NET\_SDK\_CLIENTINFO,channel number starts from 0

*dwBrightValue*

[out] brightness,range[0,255] *dwContrastValue*

[out] contrast,range[0,255] *dwSaturationValue*

[out] saturation,range[0,255] *dwHueValue*

[out] gray scale,range[0,255]

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_GetVideoEffect**

Set the display parameter of the video.

BOOL NET\_SDK\_GetVideoEffect( LONG *lUserID*,

LONG *lChannel*, DWORD *\*pBrightValue*,

DWORD *\*pContrastValue*, DWORD *\*pSaturationValue*, DWORD *\*pHueValue*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] return value of NET\_SDK\_CLIENTINFO,channel number starts from 0

*pBrightValue*

[out] pointer to brightness,range[0,255] *pContrastValue*

[out] pointer to contrast,range[0,255] *pSaturationValue*

[out] pointer to saturation,range[0,255] *pHueValue*

[out] pointer to gray scale,range[0,255]

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_SetVideoEffect\_Ex**

Set the display parameter of the video.

BOOL NET\_SDK\_SetVideoEffect\_Ex( LONG *lUserID*,

LONG *lChannel*, DWORD *dwBrightValue*,

DWORD *dwContrastValue*, DWORD *dwSaturationValue*, DWORD *dwHueValue*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *lChannel*

[in] the return value of NET\_SDK\_CLIENTINFO. The channel number starts from 0

*dwBrightValue*

[out] Brightness: the return value range of NET\_SDK\_SetVideoEffect\_Ex : [minValue,maxValue]

*dwContrastValue*

[out] Contrast: the return value range of NET\_SDK\_SetVideoEffect\_Ex: [minValue,maxValue]

*dwSaturationValue*

[out] Saturation: the return value range of NET\_SDK\_SetVideoEffect\_Ex : [minValue,maxValue]

*dwHueValue*

[out] Hue: the return value range of NET\_SDK\_SetVideoEffect\_Ex :[minValue,maxValue]

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

**Android Interface**

The corresponding Android interface：NO

Client SDK Instructions

**typedef struct {**

**unsigned int** **minValue; // the minimum value**

**unsigned int**

**// the maximum value unsigned int**

**// the current value unsigned int**

**maxValue;**

**curValue;**

**defaultValue;**

**// the default value }NET\_SDK\_IMAGE\_EFFECT\_T;**

**NET\_SDK\_GetVideoEffect\_Ex**

Get the display parameter of video.

BOOL NET\_SDK\_GetVideoEffect\_Ex( LONG *lUserID*,

LONG *lChannel*, NET\_SDK\_IMAGE\_EFFECT\_T *\*pBrightValue*,

NET\_SDK\_IMAGE\_EFFECT\_T *\*pContrastValue*, NET\_SDK\_IMAGE\_EFFECT\_T *\*pSaturationValue*, NET\_SDK\_IMAGE\_EFFECT\_T *\*pHueValue*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login(). *lChannel*

[in] the return value of NET\_SDK\_CLIENTINFO. The channel number starts from 0.

*pBrightValue*

[out] a pointer to brightness *pContrastValue*

[out] a pointer to contrast *pSaturationValue*

[out] a pointer to saturation *pHueValue*

[out] a pointer to hue

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

**Android Interface**

The corresponding Android interface：NO

Client SDK Instructions

**NET\_SDK\_GetSDKBuildVersion**

get version and build information of SDK

DWORD NET\_SDK\_GetSDKBuildVersion( );

**Return Values**

Version and build information of SDK.Two high bytes are the version:25~32 bits are main version,17~24 bits are minor version;two low bytes are build information. Take 0x01000101 for example:version is 1.0,build number is 0101.

Client SDK Instructions

**NET\_SDK\_GetSDKVersion**

get information of SDK version

DWORD NET\_SDK\_GetSDKVersion( );

**Return Values**

Version of SDK.Two high bytes are the version:25~32 bits are main version,17~24 bits are minor version;two low bytes are build information. Take 0x01000101 for example:version is 1.0,build number is 0101.

Client SDK Instructions

**NET\_SDK\_SetLogToFile**

enable writing log file

BOOL NET\_SDK\_SetLogToFile( BOOL *bLogEnable*,

char *\*strLogDir*, BOOL *bAutoDel*

);

**Parameters**

*bLogEnable*

[in] whether enable the function of writing log,default value is FALSE

*strLogDir*

[in] directory of log file,default directory is "C:\\SdkLog\\"

*bAutoDel*

[in] whether delete excess file count,default value is TRUE

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Directory of log file must be absolute path,and ends with "\\",for example:"C:\\SdkLog\\".Advise user to create file manually,if no specified path,adopt default path "C:\\SdkLog\\". This interface can be called many times to create log files, and support creating 10 files at most.When assign bAutoDel TRUE,system will delete excess file

automatically.New directory will be valid in writing file when changing directory or next time to write file.

Client SDK Instructions

**NET\_SDK\_GetErrorMsg**

return the last error code message

char\* NET\_SDK\_GetErrorMsg( LONG *\*pErrorNo*

);

**Parameters**

*pErrorNo*

[out] pointer to the value of error code

**Return Values**

return value is pointer to error code information. error message has two main types,error message of network communication library and error message of soft and hard decoding library,list the first type as follows:

**error message of network communication library**

**type of errors**

NET\_SDK\_SUCCESS

NET\_SDK\_PASSWORD\_ERROR

NET\_SDK\_NOENOUGH\_AUTH

NET\_SDK\_NOINIT

NET\_SDK\_CHANNEL\_ERROR

NET\_SDK\_OVER\_MAXLINK

NET\_SDK\_LOGIN\_REFUSED

NET\_SDK\_VERSION\_NOMATCH

NET\_SDK\_NETWORK\_FAIL\_CONNECT

NET\_SDK\_NETWORK\_NOT\_CONNECT

NET\_SDK\_NETWORK\_SEND\_ERROR

NET\_SDK\_NETWORK\_RECV\_ERROR

NET\_SDK\_NETWORK\_RECV\_TIMEOUT

NET\_SDK\_NETWORK\_ERRORDATA

NET\_SDK\_ORDER\_ERROR

NET\_SDK\_OPER\_BY\_OTHER

NET\_SDK\_OPER\_NOPERMIT

NET\_SDK\_COMMAND\_TIMEOUT

NET\_SDK\_ERROR\_SERIALPORT

NET\_SDK\_ERROR\_ALARMPORT

NET\_SDK\_PARAMETER\_ERROR

NET\_SDK\_CHAN\_EXCEPTION

NET\_SDK\_NODISK

**error value**

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

**i**

no error

user's name or passw

no right for this ope

SDK is not initialize

error of channel num

the client connected

SDK login is refuse

version doesn't matc

failed to connect to

network isn't conne

failed to send data t

failed to receive the

timeout when receiv

send illegal data to

the called order erro

operation method is

the privileged user i

DVR command tim

error of serial port n

error of alarm port n

parameter error

server's channel is i

no hard disk

NET\_SDK\_ERROR\_DISKNUM

NET\_SDK\_DISK\_FULL

NET\_SDK\_DISK\_ERROR

NET\_SDK\_NOSUPPORT

NET\_SDK\_BUSY

NET\_SDK\_MODIFY\_FAIL

NET\_SDK\_PASSWORD\_FORMAT\_ERROR

NET\_SDK\_DISK\_FORMATING

NET\_SDK\_DVR\_NORESOURCE

NET\_SDK\_DVR\_OPRATE\_FAILED

NET\_SDK\_OPEN\_HOSTSOUND\_FAIL

NET\_SDK\_DVR\_VOICEOPENED

NET\_SDK\_TIME\_INPUTERROR

NET\_SDK\_NOSPECFILE

NET\_SDK\_CREATEFILE\_ERROR

NET\_SDK\_FILEOPENFAIL

NET\_SDK\_OPERNOTFINISH

NET\_SDK\_GETPLAYTIMEFAIL

NET\_SDK\_PLAYFAIL

NET\_SDK\_FILEFORMAT\_ERROR

NET\_SDK\_DIR\_ERROR

NET\_SDK\_ALLOC\_RESOURCE\_ERROR

NET\_SDK\_AUDIO\_MODE\_ERROR

NET\_SDK\_NOENOUGH\_BUF

NET\_SDK\_CREATESOCKET\_ERROR

23 hard disk no. error

24 server hark disk is f

25 server hard disk erro

26 server does not supp

27 server is busy

28 failed to modify in t

29 the password inputt

30 hard disk is formatt

31 DVR no resources

32 DVR failed to opea

33 failed open PC voic

34 server voice dialogu

35 time input is not cor

36 there is no appointe

37 failed to create a fil

38 faile to open a file

39 the last operation is

40 faile to get the curre

41 failed to play

42 the file input format

43 path error

44 resources allotting e

45 display card mode e

46 buffer is not enough

47 establish SOCKET

NET\_SDK\_SETSOCKET\_ERROR

NET\_SDK\_MAX\_NUM

NET\_SDK\_USERNOTEXIST

NET\_SDK\_WRITEFLASHERROR

NET\_SDK\_UPGRADEFAIL

NET\_SDK\_CARDHAVEINIT

NET\_SDK\_PLAYERFAILED

NET\_SDK\_MAX\_USERNUM

NET\_SDK\_GETLOCALIPANDMACFAIL

NET\_SDK\_NOENCODEING

NET\_SDK\_IPMISMATCH

NET\_SDK\_MACMISMATCH

NET\_SDK\_UPGRADELANGMISMATCH

NET\_SDK\_MAX\_PLAYERPORT

NET\_SDK\_NOSPACEBACKUP

NET\_SDK\_NODEVICEBACKUP

NET\_SDK\_PICTURE\_BITS\_ERROR

NET\_SDK\_PICTURE\_DIMENSION\_ERROR

NET\_SDK\_PICTURE\_SIZ\_ERROR

NET\_SDK\_LOADPLAYERSDKFAILED

NET\_SDK\_LOADPLAYERSDKPROC\_ERROR

NET\_SDK\_LOADDSSDKFAILED

NET\_SDK\_LOADDSSDKPROC\_ERROR

NET\_SDK\_DSSDK\_ERROR

NET\_SDK\_VOICEMONOPOLIZE

48 set SOCKET error

49 the max number

50 user doest not exit

51 wirte FLASH error

52 failed to upgrade DV

53 the decode card is in

54 player failed

55 the max user no.

failed to get the IP a 56 end or physical add

57 the channel is not co

58 IP address not matc

59 MAC address not m

60 the language of upg

61 reach to the max pla

62 no enough space to

63 no backup device

64 the bits of picture n

65 the dimension is ov

66 the size of picture is

67 failed to load player

68 not find some funct

69 failed to load DsSD

70 not find some funct

71 failed to call functio

72 voice card is monop

NET\_SDK\_JOINMULTICASTFAILED 73

NET\_SDK\_CREATEDIR\_ERROR 74

NET\_SDK\_BINDSOCKET\_ERROR 75

NET\_SDK\_SOCKETCLOSE\_ERROR 76

NET\_SDK\_USERID\_ISUSING 77

NET\_SDK\_PROGRAM\_EXCEPTION 78

NET\_SDK\_WRITEFILE\_FAILED 79

NET\_SDK\_FORMAT\_READONLY 80

NET\_SDK\_WITHSAMEUSERNAME 81

NET\_SDK\_DEVICETYPE\_ERROR 82

NET\_SDK\_LANGUAGE\_ERROR 83

NET\_SDK\_PARAVERSION\_ERROR 84

NET\_SDK\_FILE\_SUCCESS 85

NET\_SDK\_FILE\_NOFIND 86

NET\_SDK\_NOMOREFILE 87

NET\_SDK\_FILE\_EXCEPTION 88

NET\_SDK\_TRY\_LATER 89

NET\_SDK\_DEVICE\_OFFLINE 90

NET\_SDK\_CREATEJPEGSTREAM\_FAIL 91

NET\_SDK\_USER\_ERROR\_NO\_USER 92

NET\_SDK\_USER\_ERROR\_USER\_OR\_PASSWORD\_IS\_NULL 93

NET\_SDK\_USER\_ERROR\_ALREDAY\_LOGIN 94

NET\_SDK\_USER\_ERROR\_SYSTEM\_BUSY 95

NET\_SDK\_DEVICE\_NOT\_SUPPROT 96

NET\_SDK\_USER\_ERROR\_SYSTEM\_NO\_READY 97

failed join to multic

failed to create log f

failed to bind socke

socket is closed

the user ID is opera

sdk program except

write file failed

failed to format read

there is same userna

device type no matc

language no match

soft version no matc

file has been created

file isn't found

there is no more file

file exception

Try again later

Device offline

Failed to create JPE

No such user!

No username or pas

The user has been lo

The device is busy.

The device don not

Do not complete ge

NET\_SDK\_CHANNEL\_OFFLINE

NET\_SDK\_GETREADYINFO\_FAIL

NET\_SDK\_NORESOURCE

NET\_SDK\_DEVICE\_QUERYSYSTEMCAPS\_FAIL

NET\_SDK\_INBUFFER\_TOSMALL

NET\_SDK\_NO\_PASSWORD\_STRENGTH

98 Camera is offline.

99 It fails to get device

100 SDK resources is no

101 The device fails to g

102 The input buffer are

103 The password stren

**Remarks**

Get error number through function NET\_SDK\_GetErrorMsg

**See Also**

[NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_Login**

user to register device

LONG NET\_SDK\_Login( char

WORD char char

[NET\_SDK\_DEVICEINFO](#_page_681_0) );

*\*sDVRIP*, *wDVRPort*, *\*sUserName*, *\*sPassword*,

*lpDeviceInfo*

**Parameters**

*sDVRIP*

[in] device IP address *wDVRPort*

[in] number of device port *sUserName*

[in] user name for login *sPassword*

[in] user password *lpDeviceInfo*

[out] device information

**Return Values**

-1 means failure and other value is the returned ID from user.The ID is unique. Later operations on device realize through this ID. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Device permits 32 registered names,and supports 128 users to register at the same time.

**See Also**

[NET\_SDK\_Logout](#_page_141_0)

Client SDK Instructions

**NET\_SDK\_Logout**

user to logout

BOOL NET\_SDK\_Logout( LONG *lUserID*

);

**Parameters**

*lUserID*

[in] user ID,return value of NET\_SDK\_Login

**Return Values**

TRUE means success; FALSE means failure.To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_LoginEx**

user to register device

NET\_SDK\_API LONG CALL\_METHOD NET\_SDK\_LoginEx( char *\*sDVRIP*,

WORD *wDVRPort*, char *\*sUserName*, char *\*sPassword*,

[NET\_SDK\_DEVICEINFO](#_page_681_0) *lpDeviceInfo*, NET\_SDK\_CONNECT\_TYPE *eConnectType*, const char *\*sDevSN*

);

**Parameters**

*sDVRIP*

[in] device IP address or P2P Server address *wDVRPort*

[in] number of device port or P2P Server port *sUserName*

[in] user name for login *sPassword*

[in] user password *lpDeviceInfo*

[out] device informantion *eConnectType*

[in] Connect type,refer to NET\_SDK\_CONNECT\_TYPE:

**Type** **Value** NET\_SDK\_CONNECT\_TCP 0 NET\_SDK\_CONNECT\_NAT 1 NET\_SDK\_CONNECT\_NAT20 2

**Description** TCP connection NAT1.0 connection NAT2.0 connection

*sDevSN*

[in] Serial number of P2P. If TCP is connected, ignore this parameter

**Return Values**

-1 means failure and other value is the returned ID from user.The ID is unique. Later operations on device realize through this ID. To get error information,please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

Device permits 32 registered names,and supports 128 users to register at the same time.

**See Also**

[NET\_SDK\_Logout](#_page_141_0)

Client SDK Instructions

**NET\_SDK\_SetNat2Addr**

set the p2p2.0 address

LONG NET\_SDK\_SetNat2Addr( char *\**sServerAddr, WORD *wDVRPort*, );

**Parameters**

*sServerAddr*

[in] device IP address （it is the p2p server address （c2020.autonat.com））

*wDVRPort*

[in] p2p port

**Return Values**

TRUE means success; FALSE means failure.To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

NET\_SDK\_LoginEx

Client SDK Instructions

**NET\_SDK\_MakeKeyFrame**

one key frame from main code stream dynamically

BOOL NET\_SDK\_MakeKeyFrame( LONG *lUserID*,

LONG *lChannel* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is set to reset I frame.According to the preview parameter(NET\_SDK\_CLIENTINFO) type main code stream or sub code stream calls NET\_SDK\_MakeKeyFrame or NET\_SDK\_MakeKeyFrameSub to realize resetting I frame.

**See Also**

[NET\_SDK\_MakeKeyFrameSub](#_page_154_0)

Client SDK Instructions

**NET\_SDK\_MakeKeyFrameEx**

one key frame from main code stream dynamically

BOOL NET\_SDK\_MakeKeyFrameEx( LONG *lUserID*,

LONG unsigned );

*lChannel*，

int *streamType*

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0

*streamType*

[in] stream type

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

This interface is set to reset I frame.According to the preview parameter(NET\_SDK\_CLIENTINFO) type main code stream or sub code stream calls NET\_SDK\_MakeKeyFrame or NET\_SDK\_MakeKeyFrameSub to realize resetting I frame.

**See Also**

[NET\_SDK\_MakeKeyFrameSub](#_page_154_0)

Client SDK Instructions

**NET\_SDK\_MakeKeyFrameSub**

one key frame from sub code stream dynamically

BOOL NET\_SDK\_MakeKeyFrameSub( LONG *lUserID*,

LONG *lChannel* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number starts from 0

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is set to reset I frame.According to the preview parameter(NET\_SDK\_CLIENTINFO) type main code stream or sub code stream calls NET\_SDK\_MakeKeyFrame or NET\_SDK\_MakeKeyFrameSub to realize resetting I frame.

**See Also**

[NET\_SDK\_MakeKeyFrame](#_page_149_0)

Client SDK Instructions

**NET\_SDK\_LivePlay**

real time preview

LONG NET\_SDK\_LivePlay( LONG [LPNET\_SDK\_CLIENTINFO](#_page_676_0) [LIVE\_DATA\_CALLBACK](#_page_197_0) void

);

*lUserID*, *lpClientInfo*, *fLiveDataCallBack*, *\* pUser*

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *lpClientInfo*

[in] preview parameter *fLiveDataCallBack*

[in] preview data callback parameter,default value is NULL

*\* pUser*

[in] pointer to user,default value is NULL

**Return Values**

-1 means failure and other value is parameter of handle of function NET\_SDK\_StopLivePlay. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

After calling this interface successfully,if need to capture real time code stream data,call NET\_SDK\_SetLiveDataCallBack to register the callback

function for capturing code stream data,and access the code stream data in the callback function.

**See Also**

[NET\_SDK\_StopLivePlay](#_page_159_0)

Client SDK Instructions

**NET\_SDK\_StopLivePlay**

stop preview

BOOL NET\_SDK\_StopLivePlay( LONG *lLiveHandle*

);

**Parameters**

*lLiveHandle*

[in] handle for preview,return value of NET\_SDK\_LivePlay

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**DRAW\_FUN\_CALLBACK**

data callback when preview

void \*DRAW\_FUN\_CALLBACK( LONG *lLiveHandle*, HDC *hDC*,

void *\* pUser* );

**Parameters**

*lLiveHandle*

[in] preview interface handle *hDC*

[in] device context handle *\*pUser*

[in] pointer to user information

**Return Values**

None. To get error code,please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_LivePlay\_Ex**

Real-time Preview

LONG NET\_SDK\_LivePlayEx( LONG *lUserID*, [LPNET\_SDK\_CLIENTINFO](clbr://internal.invalid/book/struct/LPNET_SDK_CLIENTINFO.htm)

LIVE\_DATA\_CALLBACK\_EX void *\* pUser* );

*lpClientInfo*, *fLiveDataCallBack*,

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *lpClientInfo*

[in] preview parameters *fLiveDataCallBack*

[in] preview data callback parameters, the default value is NULL .

*\* pUser*

[in] user pointer, the default value is NULL.

**Return Values**

-1 means failure; other values is the handles of the functions, like NET\_SDK\_StopLivePlay. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

Having sucessfully called up this interface, call up the callback function of stream data captured through the registeration of the interface

(NET\_SDK\_SetLiveDataCallBackEx) to get the real-time stream data.

**See Also**

[NET\_SDK\_StopLivePlay](#_page_159_0) [NET\_SDK\_GetLivePlayerIndex](clbr://internal.invalid/book/NET_SDK_GetLivePlayerIndex.htm)

Client SDK Instructions

**NET\_SDK\_SupportStreamNum**

get the number of streams which support live

unsigned int LONG

LONG );

NET\_SDK\_SupportStreamNum( *lUserID*, *lChannel*

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *lChannel*

[in] the index of channel，from 0

**Return Values**

return the number of streams which support live。To get error information, please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_RegisterDrawFun**

overlapping picture of character and image when preview or playback

BOOL NET\_SDK\_RigisterDrawFun( LONG *lLiveHandle*, [DRAW\_FUN\_CALLBACK](#_page_161_0) *fDrawFun*, void *\*pUser*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *fDrawFun*

[in] callback function for image *pUser*

[in] pointer to user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is mainly used to register callback function,and get current surface device context. User can draw or write on this DC just like on the window's client DC,but this DC is Off-Screen surface DC in player DirectDraw,not window's client DC .

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_SetPlayerBufNumber**

set frame buffer area count of broadcast library

BOOL NET\_SDK\_SetPlayerBufNumber( LONG *lLiveHandle*,

DWORD *dwBufNum* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *dwBufNum*

[in] the maximum frame count in buffer area when playing one frame by one frame,range[1,50],default value is 15 in SDK

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is called to adjust network delay and play fluency.value of dwBufNum more large,fluency more well,relative delay more large and vice versa. But when network isn't enough well,lost frame makes effect on fluency of broadcast.If current stream is complex,assign buffer frame count equal to or greater than 6 to ensure audio and video effect. This function follows NET\_SDK\_LivePlay closely,because it's invalid after playing image .

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_OpenSound**

open sound in monopolistic sound card mode

BOOL NET\_SDK\_OpenSound( LONG *lLiveHandle*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

If current mode is sharing,calling this interface returns failure.The monopolistic mode just opens one channel to playback, when in turn to open multichannel,just opens the last one.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0) [NET\_SDK\_CloseSound](#_page_178_0)

Client SDK Instructions

**NET\_SDK\_Volume**

adjust play volume

BOOL NET\_SDK\_Volume( LONG *lLiveHandle*, WORD *wVolume*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *wVolume*

[in] volumeless, range[0,0xffff]

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_CloseSound**

close sound under monopolistic sound card mode

BOOL NET\_SDK\_CloseSound( );

**Return Values**

TRUE means success; FALSE means failure. to get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_OpenSound](#_page_174_0)

Client SDK Instructions

**NET\_SDK\_SetLiveDataCallBack**

set preview data callback

BOOL NET\_SDK\_SetLiveDataCallBack( LONG *lLiveHandle*, [LIVE\_DATA\_CALLBACK](#_page_197_0) *fLiveDataCallBack*, void *\*pUser*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *fLiveDataCallBack*

[in] callback function of code stream *pUser*

[in] user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This function includes start and stop operation on user dispose captured data by SDK,when assigning callback function fLiveDataCallBack other value except NULL, start callback and disposing data,when assigning NULL,stop callback and disposing data.The first package in callback is a file head with 40 bytes,for the later usage in decoding,the next package which to call is compressed code stream.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_SaveLiveData**

save real time preview data

BOOL NET\_SDK\_SaveLiveData( LONG *lLiveHandle*,

char *\*sFileName* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *\*sFileName*

[in] pointer to file directory

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0) [NET\_SDK\_StopSaveLiveData](#_page_188_0)

Client SDK Instructions

**NET\_SDK\_SetLiveDataCallBackEx**

Set the callback of preview data.

BOOL NET\_SDK\_SetLiveDataCallBackEx( LONG *lLiveHandle*,

[LIVE\_DATA\_CALLBACK\_EX](clbr://internal.invalid/book/LIVE_DATA_CALLBACK_EX.htm) *fLiveDataCallBack*, void *\*pUser*

);

**Parameters**

*lLiveHandle*

[in] the return value of NET\_SDK\_LivePlay() *fLiveDataCallBack*

[in] callback function of stream data *pUser*

[in] user data

**Return Values**

TRUE means success; FALSE means failure. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

This function includes start and stop operation on user dispose captured data by SDK,when assigning callback function fLiveDataCallBack other value except NULL, start callback and disposing data,when assigning NULL,stop callback and disposing data.The first package in callback is a file head with 40 bytes,for the later usage in decoding,the next package which to call is compressed code stream.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_StopSaveLiveData**

stop data capture

BOOL NET\_SDK\_StopSaveLiveData( LONG *lLiveHandle*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0) [NET\_SDK\_SaveLiveData](#_page_183_0)

Client SDK Instructions

**YUV\_DATA\_CALLBACK**

YUV data cabllback after decoding the captured real-time data

void YUV\_DATA\_CALLBACK( POINTERHANDLE

[DECODE\_FRAME\_INFO](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/DECODE_FRAME_INFO.htm) void *\* pUser* );

*lLiveHandle*, *frameInfo*,

**Parameters**

*lLiveHandle*

[in] the handle of real-time preview *frameInfo*

[in] YUV data after decoding the frame of real-time preview data

*\*pUser*

[in] the pointer of user information

**Return Values**

No return value. To get error information, please call [NET\_SDK\_GetLastError](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/NET_SDK_GetlastError.htm)

**Android Interface**

Client SDK Instructions

**NET\_SDK\_SetYUVCallBack**

Configure the YUV data cabllback after decoding video.

BOOL NET\_SDK\_SetYUVCallBack( LONG *lLiveHandle*,

[YUV\_DATA\_CALLBACK](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/YUV_DATA_CALLBACK.htm) *fYuvCallBack*, void *\*pUser*

);

**Parameters**

*lLiveHandle*

[in] the return value of NET\_SDK\_LivePlay() *fYuvCallBack*

[in] YUV stream data callback function *pUser*

[in] user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/NET_SDK_GetlastError.htm)

**Remarks**

This function includes starting and stopping processing data captured by SDK . When the callback function (fYuvCallback) set to any vaule except null, it means callback and processing data; when it set to null, it means stopping callback and processing data. The callback data are the YUV data after decoding.

**See Also**

[NET\_SDK\_LivePlay](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/NET_SDK_LivePlay.htm)

**Android Interface**

Client SDK Instructions

**LIVE\_DATA\_CALLBACK**

data callback when capture data in living preview

void LIVE\_DATA\_CALLBACK( LONG

[NET\_SDK\_FRAME\_INFO](#_page_688_0) BYTE

void );

*lLiveHandle*, *frameInfo*, *\*pBuffer*,

*\* pUser*

**Parameters**

*lLiveHandle*

[in] real time preview handle *frameInfo*

[in] preview data frame information in real time *\*pBuffer*

[in] pointer to buffer area *\*pUser*

[in] pointer to user information

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**LIVE\_DATA\_CALLBACK\_Ex**

Data callback when capturing the real-time preview data.

void LIVE\_DATA\_CALLBACK\_Ex( LONG *lLiveHandle*, UINT *dataType*, BYTE *\*pBuffer*, UINT *dataLen*, void *\* pUser*

);

**Parameters**

*lLiveHandle*

[in] real-time preview handle *dataType*

[out] data frame type, see DD\_FRAME\_TYPE *\*pBuffer*

[out] a pointer to the buffer, the contents is NET\_SDK\_FRAME\_INFO + FrameData *dataLen*

[out] the data length of the buffer *\*pUser*

[out] a pointer to a user

**Return Values**

No return value. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_CapturePicture**

capture data one frame by one frame and store to be BMP file.

BOOL NET\_SDK\_CapturePicture( LONG *lLiveHandle*,

char *\*sPicFileName* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay() *sPicFileName*

[in] directory of storing .BMP image,the length of directory is less than or equals to 256 bytes

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to capture present frame of decoding to be .BMP image.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0) [NET\_SDK\_CapturePicture\_Other](#_page_203_0)

Client SDK Instructions

**NET\_SDK\_CapturePicture\_Other**

capture data one frame by one frame and store to be BMP file.

BOOL NET\_SDK\_CapturePicture\_Other( LONG *lUserID*,

LONG *lChannel*, char *\*sPicFileName* );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number *sPicFileName*

[in] directory of storing .BMP image,the length of directory is less than or equals to 256 bytes

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to capture certain user ID and channel frame of decoding to be .BMP image.

**See Also**

[NET\_SDK\_CapturePicture](#_page_201_0)

Client SDK Instructions

**NET\_SDK\_CaptureJPEGData\_V2**

Capture data in JPEG format.

BOOL NET\_SDK\_CaptureJPEGData\_V2( LONG *lUserID*,

LONG *lChannel*,

char *\*sJpegPicBuffer*, DWORD *dwPicSize*,

LPDWORD *lpSizeReturned* );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number *sJpegPicBuffer*

[out] JPEG data buffer *dwPicSize*

[in] sJpegPicBuffer buffer size *lpSizeReturned*

[out] JPEG data size

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to capture jpeg data,do not need to open the stream. Only applicable to specific firmware.

**See Also**

[NET\_SDK\_CaptureJPEGFile\_V2](#_page_208_0)

Client SDK Instructions

**NET\_SDK\_CaptureJPEGFile\_V2**

Capture data in JPEG format and store to be JPEG file.

BOOL NET\_SDK\_CaptureJPEGFile\_V2( LONG *lUserID*,

LONG *lChannel*, char *\*sPicFileName* );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number *\*sPicFileName*

[in] save the file path

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call[NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to capture jpeg data and store,do not need to open the stream. Only applicable to specific firmware

**See Also**

[NET\_SDK\_CaptureJPEGData\_V2](#_page_205_0)

Client SDK Instructions

**NET\_SDK\_CaptureJPEGPict ure**

Capture data in JPEG format and store to be JPEG file.

BOOL NET\_SDK\_CaptureJPEGFile\_V2( LONG *lUserID*,

LONG *lChannel*, LPNET\_SDK\_JPEGPARA lpJpegPara, char \*sJpegPicBuffer ,

DWORD dwPicSize, LPDWORD lpSizeReturned );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number *\*LPNET\_SDK\_JPEGPARA*

[in] a command type *sJpegPicBuffer*

[in]JPEG data buffer *dwPicSize*

[in]sJpegPicBuffer buffer size *lpSizeReturned*

[in]JPEG data size

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to capture jpeg data and store,do not need to open the stream. Only applicable to specific firmware

**See Also**

[NET\_SDK\_CaptureJPEGData\_V2](#_page_205_0)

**Android Interface**

The corresponding Android interface：no

**Client SDK Instructions**

**NET\_SDK\_RemoteSnap**

Caputre images by controlling devices remotely. The images are saved in the device end.(only N9000 availabe）

BOOL NET\_SDK\_RemoteSnap( LONG *lUserID*,

int *lChannel*, );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *lChannel*

[in] Channel number（starting with 0）

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](mk:@MSITStore:C:UsersAdministratorDesktop%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C.CHM::/NET_SDK_GetlastError.htm)

Client SDK Instructions

**typedef struct \_net\_sdk\_image\_sreach**

{

DWORD dwChannel;//Snapshot channel

（starting with 0）

DD\_TIME StartTime; //time DD\_TIME StopTime; //time

DWORD pageIndex;//Page number DWORD pageSize;//Page IMAGE\_SORT\_TYPE sort; //return

unsigned char resv[8]; }NET\_SDK\_IMAGE\_SREACH;

typedef struct \_net\_sdk\_image\_ {

DWORD dwChannel; //Snapshot cahnnel DWORD dwImageType; //Snapshot type

IMAGE\_EVENT\_TYPE

DD\_TIME captureTime;//Snapshot time DWORD totalNum;//Totol number unsigned char resv[8];

}NET\_SDK\_IMAGE;

**NET\_SDK\_SearchPictures**

Get the image list of the remote device.( only N9000 device availabe)

BOOL NET\_SDK\_SearchPictures( LONG lUserID,

NET\_SDK\_IMAGE\_SREACH sInSreachImage, LONG lInImageBufferSize,

NET\_SDK\_IMAGE \*pOutImageInfo, LONG \*pOutImageNum

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() sInSreachImage

[in] Search condition *lInImageBufferSize*

[in] the space of pOutImageInfo applied for *pOutImageInfo*

[out] Return image information *pOutImageNum*

[out] the number of returning to image information

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](mk:@MSITStore:C:UsersAdministratorDesktop%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C.CHM::/NET_SDK_GetlastError.htm)

**Remarks**

This interface can realize JPEG snapshot and save the files without enabling stream. It is only available for specified

devices.

**See Also**

[NET\_SDK\_CaptureJPEGData\_V2](mk:@MSITStore:C:UsersAdministratorDesktop%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C.CHM::/NET_SDK_CaptureJPEGData_V2.htm)

**Client SDK Instructions**

**//Image Type**

enum IMAGE\_MODE {

IMAGE\_MODE\_JPG, IMAGE\_MODE\_PNG, IMAGE\_MODE\_BMP,

};

typedef struct \_net\_sdk\_image\_info {

unsigned int imageSize;

IMAGE\_MODE imageMode; //Image format unsigned char resv[8];

}NET\_SDK\_IMAGE\_INFO;

**NET\_SDK\_DownLoadPicture**

Uploading remote images（only N9000 devices availabe）

BOOL NET\_SDK\_DownLoadPicture( LONG lUserID,

NET\_SDK\_IMAGE captureImage, NET\_SDK\_IMAGE\_INFO \*pOutImageInfo, char\* pOutBuffer,

int outBufferSize );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *captureImage*

[in] NET\_SDK\_SearchPictures returns a data of the searched image list

*pOutImageInfo*

[out] return the uploaded image information *pOutBuffer*

[out] return the image data（when the current image exists and the space requested is larger than or equal to the image size, it takes effect)

*outBufferSize*

[int] buffer area size of pOutBuffer requesting

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](mk:@MSITStore:C:UsersAdministratorDesktop%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C.CHM::/NET_SDK_GetlastError.htm)

Two conditions to return successfully：

1、When the image exsits and the size of outBufferSize is less than the image size，the return will be successful and the upper application can decode the image size from pOutImageInfo parameters.

2、When the image exsits and the size of outBufferSize is larger than or equal to the downloaded image size, the return value of pOutBuffer is image data.

Client SDK Instructions

**NET\_SDK\_FindFile**

find record file by time

LONG NET\_SDK\_FindFile(

LONG *lUserID*, LONG *lChannel*, [DD\_TIME](#_page_635_0) *lpStartTime*, [DD\_TIME](#_page_635_0) *lpStopTime* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *lChannel*

[in] channel number,start from 0 *lpStartTime*

[in] the start time of file *lpStopTime*

[in] the stop time of file

**Return Values**

-1 means failure and other value is a parameter of function NET\_SDK\_FindClose. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface specifies the type and time range of finding record file,after calling the interface,call NET\_SDK\_FindNextFile to get file information.

**See Also**

[NET\_SDK\_FindNextFile](#_page_225_0) [NET\_SDK\_FindClose](#_page_227_0)

Client SDK Instructions

**NET\_SDK\_FindNextFile**

get file information one by one

LONG NET\_SDK\_FindNextFile(

LONG *lFindHandle*, [NET\_SDK\_REC\_FILE](#_page_699_0) *lpFindData* );

**Parameters**

*lFindHandle*

[in] handle of finding file,return value of NET\_SDK\_FindFile()

*lpFindData*

[out] pointer to store file information

**Return Values**

-1 means failure and other value is current state information. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Before calling this interface,call NET\_SDK\_FindFile to get current handle for finding

**See Also**

[NET\_SDK\_FindFile](#_page_222_0)

Client SDK Instructions

**NET\_SDK\_FindClose**

close finding filename, free resource

BOOL NET\_SDK\_FindClose( LONG *lFindHandle*

);

**Parameters**

*lFindHandle*

[in] return handle of finding file in function NET\_SDK\_FindFile

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindFile](#_page_222_0)

Client SDK Instructions

**NET\_SDK\_FindRecDate**

find record file by data

LONG NET\_SDK\_FindRecDate( LONG *lUserID*

);

**Parameters**

*lUserID*

[in] user ID

**Return Values**

-1 means failure and other value is the return information of finding. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindRecDateClose](#_page_233_0) [NET\_SDK\_FindNextRecDate](#_page_231_0)

Client SDK Instructions

**NET\_SDK\_FindNextRecDate**

get record file one by one

LONG NET\_SDK\_FindNextRecDate( LONG *lFindHandle*, [DD\_DATE](#_page_551_0) *\*lpRecDate*

);

**Parameters**

*lFindHandle*

[in] handle of finding *\*lpRecDate*

[in] date of record

**Return Values**

-1 means failure and other value is the return value of finding. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindRecDate](#_page_229_0) [NET\_SDK\_FindRecDateClose](#_page_233_0)

Client SDK Instructions

**NET\_SDK\_FindRecDateClose**

close finding record file by date,free resource

BOOL NET\_SDK\_FindRecDateClose( LONG *lFindHandle*

);

**Parameters**

*lFindHandle*

[in] handle of finding

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindRecDate](#_page_229_0) [NET\_SDK\_FindNextRecDate](#_page_231_0)

Client SDK Instructions

**NET\_SDK\_FindEvent**

find record file by event

LONG NET\_SDK\_FindEvent( LONG *lUserID*,

LONG *lChannel*, DWORD *dwRecType*, [DD\_TIME](#_page_635_0) *lpStartTime*, [DD\_TIME](#_page_635_0) *lpStopTime* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *lChannel*

[in] channel number,start from 0 *dwRecType*

[in] type of event *lpStartTime*

[in] pointer to the start time of file *lpStopTime*

[in] pointer to the stop time of file

**Return Values**

-1 means failure and other value is parameter of function NET\_SDK\_FindNextEvent. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface specifies information of finding record file(by event),after calling the interface,call

NET\_SDK\_FindNextEvent to get file information. The record file which is found by event aims at the start time and stop time,so it only supports playback by time.

**See Also**

[NET\_SDK\_FindNextEvent](#_page_238_0) [NET\_SDK\_FindEventClose](#_page_240_0)

Client SDK Instructions

**NET\_SDK\_FindNextEvent**

get information of found file one by one

LONG NET\_SDK\_FindNextEvent(

LONG *lFindHandle*, [NET\_SDK\_REC\_EVENT](#_page_696_0) *\*lpRecEvent* );

**Parameters**

*lFindHandle*

[in] handle of finding file,return value of NET\_SDK\_FindEvent()

*\*lpRecEvent*

[out] pointer to store file information

**Return Values**

-1 means failure and other value is current state information. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Before calling this interface,call NET\_SDK\_FindEvent to get current handle for finding.The record file found by event aims at the start time and stop time,so only supports playback by time.

**See Also**

[NET\_SDK\_FindEvent](#_page_235_0)

Client SDK Instructions

**NET\_SDK\_FindEventClose**

close finding file by event,free resource

BOOL NET\_SDK\_FindEventClose( LONG *lFindHandle*

);

**Parameters**

*lFindHandle*

[in] handle of query

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindEvent](#_page_235_0) [NET\_SDK\_FindNextEvent](#_page_238_0)

Client SDK Instructions

**NET\_SDK\_FindTime**

find record file by time

LONG NET\_SDK\_FindTime( LONG *lUserID*, LONG *lChannel*,

[DD\_TIME](#_page_635_0) *\* lpStartTime*, [DD\_TIME](#_page_635_0) *\* lpStopTime* );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number,start from 0 *\* lpStartTime*

[in] pointer to the start time *\* lpStopTime*

[in] pointer to the stop time

**Return Values**

-1 means failure and other value is the result of finding. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindNextTime](#_page_244_0) [NET\_SDK\_FindTimeClose](#_page_246_0)

Client SDK Instructions

**NET\_SDK\_FindNextTime**

get record file one by one

LONG NET\_SDK\_FindNextTime(

LONG *lFindHandle*, [NET\_SDK\_REC\_TIME](#_page_702_0) *\*lpRecTime* );

**Parameters**

*lFindHandle*

[in] handle of finding *\*lpRecTime*

[in] time of record

**Return Values**

-1 means failure and other value is found information. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindTime](#_page_242_0) [NET\_SDK\_FindTimeClose](#_page_246_0)

Client SDK Instructions

**NET\_SDK\_FindTimeClose**

close finding record file by time,free resource

BOOL NET\_SDK\_FindTimeClose( LONG *lFindHandle*

);

**Parameters**

*lFindHandle*

[in] handle of finding

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindTime](#_page_242_0) [NET\_SDK\_FindNextTime](#_page_244_0)

Client SDK Instructions

**NET\_SDK\_PlayBackByTime**

Play back the record file by time and the main stream is requested.

LONG NET\_SDK\_PlayBackByTime( LONG *lUserID*, LONG *\*pChannels*, LONG *channelNum*,

[DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime*, HWND *\*hWnd*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *pChannels*

[in] channel number,start from 0 array for playback *channelNum*

[in] quatity of channels in array **pChannels** *\*lpStartTime*

[in] pointer to the start time of the file *\*lpStopTime*

[in] pointer to the stop time of the file *\*hWnd*

[in] window handle for playback,if null,SDK still can receive code stream data,but can't decode to display.

**Return Values**

-1 means failure and other value is parameter of NET\_SDK\_StopPlayBack. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface specifies which record file to play,after finishing calling this interface,call NET\_SDK\_SetPlayDataCallBack to register callback function and dispose the captured code stream data by itself.

**See Also**

[NET\_SDK\_PlayBackControl](#_page_251_0)

[SDK\_SetLiveDataCallBack](#_page_180_0)

[NET\_SDK\_StopPlayBack](#_page_254_0) [NET\_](#_page_180_0)

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_PlayBackControl**

control the state of playback

BOOL NET\_SDK\_PlayBackControl( LONG *lPlayHandle*,

DWORD *dwControlCode*, DWORD *dwInValue*, DWORD *\*lpOutValue*

);

**Parameters**

*lPlayHandle*

[in] play handle,return value of NET\_SDK\_PlayBackByTime *dwControlCode*

[in] command of controlling the state of playback,refer to NET\_SDK\_PLAYCTRL\_TYPE:

**Type**

NET\_SDK\_PLAYCTRL\_PAUSE

**Description**

pause

NET\_SDK\_PLAYCTRL\_FF NET\_SDK\_PLAYCTRL\_REW NET\_SDK\_PLAYCTRL\_RESUME NET\_SDK\_PLAYCTRL\_STOP NET\_SDK\_PLAYCTRL\_FRAME NET\_SDK\_PLAYCTRL\_NORMAL

NET\_SDK\_PLAYCTRL\_STARTAUDIO

NET\_SDK\_PLAYCTRL\_STOPAUDIO

fast forward rewind resume stop

play one frame normal play

enable audio,choose channel by parameter *dwInValue* of function NET\_SDK\_PlayBackControl

stop audio

adjust audio volume,realize by

NET\_SDK\_PLAYCTRL\_AUDIOVOLUMEparameter *dwInValue* of function NET\_SDK\_PlayBackControl

NET\_SDK\_PLAYCTRL\_SETPOS play progress,calculate seconds from January 1st

in 1970

*dwInValue*

[in] parameter setting,when setting playback progress this parameter means progress value; when playing this parameter means file location of resuming from break point.

*lpOutValue*

[out] got parameter,if need to get total time of current play file,this parameter meets.

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Whether assign the third parameter of this interface depends on the control command,under NET\_SDK\_PLAYSETPOS command this parameter means the playback progress; when starting the control command it means offset of current file;when its value is 0 it means to play from the starting location,if not 0 ,it means the file location of resuming from break point.

The fourth parameter means the parameter got from current control command operation.

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_StopPlayBack**

stop record file playback

BOOL NET\_SDK\_StopPlayBack( LONG *lPlayHandle*

);

**Parameters**

*lPlayHandle*

[in] handle for playback,return value of NET\_SDK\_PlayBackByTime

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**PLAY\_DATA\_CALLBACK**

data callback when finding file and playback

void \*PLAY\_DATA\_CALLBACK( LONG

[NET\_SDK\_FRAME\_INFO](#_page_688_0) BYTE

void );

*lPlayHandle*, *frameInfo*, *\*pBuffer*, *\*pUser*

**Parameters**

*lPlayHandle*

[in] play handle *frameInfo*

[in] file data playback code stream frame information *\*pBuffer*

[in] buffer pointer to find files and playback *\* pUser*

[in] pointer to user information

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_PlayBackByTimeEx**

Play back the record files by time.

LONG NET\_SDK\_PlayBackByTimeEx( LONG *lUserID*,

LONG *\*pChannels*, LONG *channelNum*,

[DD\_TIME](clbr://internal.invalid/book/struct/DD_TIME.htm) *\*lpStartTime*, [DD\_TIME](clbr://internal.invalid/book/struct/DD_TIME.htm) *\*lpStopTime*,

HWND *\*hWnds* BOOL *bFirstStream* );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *pChannels*

[in] the channel number group of playback *channelNum*

[in] the channel quantities of pChannels *\*lpStartTime*

[in] a pointer to start time of the file *\*lpStopTime*

[in] a pointer to end time of the file *\*hWnds*

[in] the handle of playback window, if it is null, SDK still can receive the stream data but not decoding.

*bFirstStream*

[in] Whether to play back the main stream. false is sub-stream.

**Return Values**

-1 means failure; other vallues is the parameters of the functions, like NET\_SDK\_StopPlayBack. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

**This interface specifies the record files needed to play. After successfully call up this interface, please register callback function by the interface of NET\_SDK\_SetPlayDataCallBack to capture the recording stream data and** dispose the captured code stream data by itself.

**See Also**

[NET\_SDK\_PlayBackControl](#_page_251_0) [NET\_SDK\_StopPlayBack](#_page_254_0)

[NET\_SDK\_SetLiveDataCallBack](#_page_180_0) [NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_SetPlayDataCallBack**

register callback function,capture record data

BOOL NET\_SDK\_SetPlayDataCallBack( LONG *lPlayHandle*,

[PLAY\_DATA\_CALLBACK](#_page_256_0) *fPlayDataCallBack*, void *\*pUser*

);

**Parameters**

*lPlayHandle*

[in] play handle,return value of NET\_SDK\_PlayBackByTime

*fPlayDataCallBack*

[in] callback function of record data *pUser*

[in] user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This function includes start and stop operation on user dispose captured data by SDK,when assigning callback function fLiveDataCallBack other value except NULL, start callback and disposing data,if assign NULL,stop callback and disposing data.

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_SetPlayYUVCallBack**

(only in windows now)register callback function,capture record data.it can receive YUV data from the callback function,you should return quickly if you need to process the data.

BOOL NET\_SDK\_SetPlayYUVCallBack( LONG *lPlayHandle*,

[PLAY\_YUV\_DATA\_CALLBACK](#_page_256_0) *fYuvDataCallBack*, void *\*pUser*

);

**Parameters**

*lPlayHandle*

[in] play handle,return value of NET\_SDK\_PlayBackByTime

*fYuvDataCallBack*

[in] callback function of record data *pUser*

[in] user data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This function includes start and stop operation on user dispose captured data by SDK,when assigning callback function *fYuvDataCallBack* other value except NULL, start callback and disposing data,if assign NULL,stop callback and disposing data.

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_PlayBackSaveData**

capture record data of playback and store them into file

BOOL NET\_SDK\_PlayBackSaveData( LONG *lPlayHandle*,

LONG *lChannel*, char *\*sFileName* );

**Parameters**

*lPlayHandle*

[in] play handle,return value of NET\_SDK\_PlayBackByTime

*lChannel*

[in] channel number,start from 0 *\*sFileName*

[in] directory of storing data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_StopPlayBackSave**

stop saving record data

BOOL NET\_SDK\_StopPlayBackSave( LONG *lPlayHandle*,

LONG *lChannel* );

**Parameters**

*lPlayHandle*

[in] handle for play,return value of NET\_SDK\_PlayBackByTime

*lChannel*

[in] channel number,start from 0

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_GetPlayBackOsdTime**

get OSD time when playback

BOOL NET\_SDK\_GetPlayBackOsdTime( LONG *lPlayHandle*, [DD\_TIME](#_page_635_0) *lpOsdTime*

);

**Parameters**

*lPlayHandle*

[in] player handle,return value of NET\_SDK\_PlayBackByTime

*\*lpOsdTime*

[out] pointer to OSD time

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_RefreshPlay**

refresh window to display playback

BOOL NET\_SDK\_RefreshPlay( LONG *lPlayHandle*

);

**Parameters**

*lPlayHandle*

[in] playback handle,return value of NET\_SDK\_PlayBackByTime

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

When pause or playback in one frame,if refresh the window,the image disappears ,at that time call this interface to display,this interface is valid just when pause and playback in one frame.

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_PlayBackCaptureFile**

capture when playback and store them to file

BOOL NET\_SDK\_PlayBackCaptureFile( LONG *lPlayHandle*,

LONG *lChannel*, char *\*sFileName* );

**Parameters**

*lPlayHandle*

[in] play handle,return value of NET\_SDK\_PlayBackByTime

*lChannel*

[in] channel number,start from 0 *\*sFileName*

[in] directory of storing picture data

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackByTime](#_page_248_0)

Client SDK Instructions

**NET\_SDK\_GetFileByTime**

download record file by time

LONG NET\_SDK\_GetFileByTime( LONG *lUserID*, LONG *lChannel*,

[DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime*, char *\*sSavedFileName* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0 *\*lpStartTime*

[in] pointer to the start time *\*lpStopTime*

[in] pointer to the stop time *\*sSavedFileName*

[in] directory of storing downloaded file to PC

**Return Values**

-1 means failure and other value is parameter of NET\_SDK\_StopGetFile. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackControl](#_page_251_0) [NET\_SDK\_StopGetFile](#_page_282_0)

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetDownloadPos**

get current progress of downloading record file

int NET\_SDK\_GetDownloadPos( LONG *lFileHandle*

);

**Parameters**

*lFileHandle*

[in] handle for downloading,return value of NET\_SDK\_GetFileByTime()

**Return Values**

-1 means failure;0-100 means the progress of downloading;100 means finish;normal range is 0-100, if the return value is 200,network is unusual. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is to get progress of downloading record file by file name.

**See Also**

[NET\_SDK\_GetFileByTime](#_page_277_0)

Client SDK Instructions

**NET\_SDK\_StopGetFile**

stop downloading record file

BOOL NET\_SDK\_StopGetFile( LONG *lFileHandle*

);

**Parameters**

*lFileHandle*

[in] handle for download,return value of NET\_SDK\_GetFileByTime

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_GetFileByTime](#_page_277_0)

Client SDK Instructions

**NET\_SDK\_GetFileByTimeEx**

Extended interface of download record file by time

LONG NET\_SDK\_GetFileByTime( LONG *lUserID*, LONG *lChannel*,

[DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime*, char *\*sSavedFileName* BOOL *bCustomFormat* BOOL *bUseCallBack*

[BACKUP\_DATA\_CALLBACK](#_page_290_0) *fBackupDataCallBack* void *\*pUser*

)

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] the channel number starts from 0 *\*lpStartTime*

[in] a pointer to the start time *\*lpStopTime*

[in] a pointer to the end time *\*sSavedFileName*

[in] the directory of storing downloaded file to PC *bCustomFormat*

[in] whether to use private protocl format. TRUE means private protocol format

*bUseCallBack*

[in] whether to use call back function，TRUE means use

*fBackupDataCallBack*

[in] call back function *\*pUser*

[in] user's pointer，default is NULL

**Return Values**

-1 means failure and other value is parameter of NET\_SDK\_StopGetFile. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackControl](#_page_251_0) [NET\_SDK\_StopGetFile](#_page_282_0)

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetFileByTimeExV2**

Extended interface of download record file by time version 2

LONG NET\_SDK\_GetFileByTimeExV2( LONG *lUserID*, LONG *lChannel*,

[DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime*, char *\*sSavedFileName* char *recFormat*

BOOL *bFirstStream* BOOL *bUseCallBack*

[BACKUP\_DATA\_CALLBACK](#_page_290_0) *fBackupDataCallBack* void *\*pUser*

)

**Parameters**

*lUserID*

[in] eturn value of NET\_SDK\_Login *lChannel*

[in] the channel number starts from 0 *\*lpStartTime*

[in] a pointer to the start time *\*lpStopTime*

[in] a pointer to the end time *\*sSavedFileName*

[in] the directory of storing downloaded file to PC *recFormat*

[in] whether to use private protocl format. 1 means private protocol format

*bFirstStream*

[in] whether to download first stream, TRUE means yes

*bUseCallBack*

[in] whether to use call back function，TRUE means use

*fBackupDataCallBack* [in] call back function

*\*pUser*

[in] user's pointer，default is NULL

**Return Values**

-1 means failure and other value is parameter of NET\_SDK\_StopGetFile. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PlayBackControl](#_page_251_0) [NET\_SDK\_StopGetFile](#_page_282_0)

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

BACKUP\_DATA\_CALLBACK

Call back the data when downloading the records.

void \*PLAY\_DATA\_CALLBACK( LONG

UINT BYTE UINT void );

*lFileHandle*, *dataType*, *\*pBuffer*, *dataLen*, *\*pUser*

**Parameters**

*lFileHandle*

[in] download file handle *dataType*

[in] data type *\*pBuffer*

[in] File download stream and frame information *dataLen*

[in] data length *\* pUser*

[in] a pointer to user information

**Return Values**

No return value. To get error information, please cal[lNET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SaveFileToUsbByTime**

Save the record to the USB of the device（support only N9000）。

BOOL NET\_SDK\_SaveFileToUsbByTime( LONG *lUserID*,

[NET\_SDK\_REC\_FILE](#_page_699_0) *\*recordFile*, USB\_BACKUP\_FORMAT *recFormat* );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login. *\*recordFile*

[in] the pointer of the record file *recFormat*

[in] 0 is avi，1 is private format

**Return Values**

FALSE means failed，TRUE means success。To get error information, please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_GetSaveFileToUsbProcess](#_page_293_0)

Client SDK Instructions

**NET\_SDK\_GetSaveFileToUsbProcess**

Get the process and status of the saving record to USB device（support only N9000）。

BOOL NET\_SDK\_GetSaveFileToUsbProcess( LONG *lUserID*, [NET\_SDK\_USB\_BACKUP\_PROCESS\_EX](#_page_775_0)

*\*pUsbBackProcess*,

unsigned int *lBuffSize*, unsigned int *\*taskCount*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login *\*pUsbBackProcess*

[in] the pointer of the results *lBuffSize*

[in] the expect number of the result *\*taskCount*

[in] the actual number of the result

**Return Values**

FALSE means failed，TRUE means success。To get error information, please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_SaveFileToUsbByTime](#_page_291_0)

Client SDK Instructions

**NET\_SDK\_LockFile**

lock record file(only support IPC)

BOOL NET\_SDK\_LockFile( LONG *lUserID*,

[NET\_SDK\_REC\_FILE](#_page_699_0) *\*pFileTolock*, LONG *fileNum*

);

**Parameters**

*lUserID*

[in] user ID *pFileTolock*

[in] pointer to lock file *fileNum*

[in] quantity of file

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_UnlockFile](#_page_297_0)

Client SDK Instructions

**NET\_SDK\_UnlockFile**

unlock record file(only support IPC)

BOOL NET\_SDK\_UnlockFile( LONG *lUserID*,

[NET\_SDK\_REC\_FILE](#_page_699_0) *\*pFileToUnlock*, LONG *fileNum*

);

**Parameters**

*lUserID*

[in] user ID *pFileToUnlock*

[in] pointer to unlock file *fileNum*

[in] quantity of files

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_LockFile](#_page_295_0)

Client SDK Instructions

**NET\_SDK\_DeleteRecFile**

delete recorded file(only support 3.0DVR)

BOOL NET\_SDK\_DeleteRecFile( LONG *lUserID*,

[NET\_SDK\_REC\_FILE](#_page_699_0) *\*pFileToUnlock*, LONG *fileNum*

);

**Parameters**

*lUserID*

[in] user ID *\*pFileToUnlock*

[in] pointer to unlock file *fileNum*

[in] number of files

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_StartDVRRecord**

start device to record manually and remotely(only support N9000)

BOOL NET\_SDK\_StartDVRRecord( LONG *lUserID*,

LONG *lChannel*, LONG *lRecordType* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0,0x00ff means all analog channels,0xff00 means all digital channels,0xffff means all analog channels and digital channels

*lRecordType*

[in] record type

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StopDVRRecord](#_page_303_0)

Client SDK Instructions

**NET\_SDK\_StopDVRRecord**

stop device record manually and remotely(only support N9000)

BOOL NET\_SDK\_StopDVRRecord( LONG *lUserID*,

LONG *lChannel* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0,0x00ff means all analog channels,0xff00 means all digital channels,0xffff means all analog channels and digital channels

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StartDVRRecord](#_page_301_0)

Client SDK Instructions

**NET\_SDK\_SetDVRMessageCallBack**

register callback function,receive alarm information from device etc.

BOOL NET\_SDK\_SetDVRMessCallBack( [NET\_MESSAGE\_CALLBACK](#_page_307_0) *fMessCallBack* void *\*pUser*

);

**Parameters**

*fMessCallBack*

[in] callback function *pUser*

[in] user information

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_SetupAlarmChan](#_page_315_0)

Client SDK Instructions

**NET\_MESSAGE\_CALLBACK**

data callback when alarm

BOOL NET\_MESSAGE\_CALLBACK( LONG *lCommand*, LONG *lUserID*, char *\*pBuf*, DWORD *dwBufLen*, void *\* pUser*

);

**Parameters**

*lCommand*

[in] command handle,refer to the list below:

**Type**

NET\_SDK\_ALARM

NET\_SDK\_RECORD

NET\_SDK\_IVM\_RULE

NET\_SDK\_TRADEINFO

NET\_SDK\_IPCCFG

**Description**

device alarm information

device record information

Intelligent behavior analysis information(reserved)

ATM trade information(reserved)

IPC information change of mixed DVR(reserved)

*lUserID*

[in] user ID

*\*pBuf*

[in] pointer to buffer area,when **lCommand** value is NET\_SDK\_ALARM **pBuf** is array of struct NET\_SDK\_ALARMINFO,when **lCommand** value is NET\_SDK\_RECORD **pBuf** is array of struct NET\_SDK\_RECORD\_STATUS

*dwBufLen*

[in] length of buffer area *\* pUser*

[in] pointer to user information

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_SetDVRMessageCallBackEx**

register callback function,receive alarm information from device etc.

BOOL NET\_SDK\_SetDVRMessageCallBackEx( [NET\_MESSAGE\_CALLBACK\_EX](#_page_312_0) *fMessCallBack* void *\*pUser*

);

**Parameters**

*fMessCallBack*

[in] callback function *pUser*

[in] user information

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_SetupAlarmChan](#_page_315_0)

Client SDK Instructions

**NET\_MESSAGE\_CALLBACK\_EX**

data callback when alarm

BOOL NET\_MESSAGE\_CALLBACK\_EX( LONG *lCommand*,

LONG *lUserID*, char *\*pBuf*, DWORD *dwBufLen*, void *\* pUser* );

**Parameters**

*lCommand*

[in] command handle,refer to the list below:

**Type**

NET\_SDK\_ALARM

NET\_SDK\_RECORD

NET\_SDK\_IVM\_RULE

NET\_SDK\_TRADEINFO

NET\_SDK\_IPCCFG

**Description**

device alarm information

device record information

Intelligent behavior analysis information(reserved)

ATM trade information(reserved)

IPC information change of mixed DVR(reserved)

*lUserID*

[in] user ID

*\*pBuf*

[in] pointer to buffer area,when **lCommand** value is NET\_SDK\_ALARM **pBuf** is array of struct [NET\_SDK\_ALARMINFO\_EX](#_page_673_0) ,when **lCommand** value is NET\_SDK\_RECORD **pBuf** is array of

struct [NET\_SDK\_RECORD\_STATUS\_EX](#_page_707_0) *dwBufLen*

[in] length of buffer area *\* pUser*

[in] pointer to user information

**Return Values**

None. To get error code, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_SetupAlarmChan**

build uploading channel for alarm,get alarm information

LONG NET\_SDK\_SetupAlarmChan( LONG *lUserID*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login

**Return Values**

-1 means failure and other value is handle parameter of NET\_SDK\_CloseAlarmChan. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Start arming and then call the interface of register callback function(just call NET\_SDK\_SetDVRMessCallBack when need)to get uploading information.

**See Also**

[NET\_SDK\_CloseAlarmChan](#_page_317_0) [NET\_SDK\_Login](#_page_138_0)

[NET\_SDK\_SetDVRMessCallBack](#_page_305_0)

Client SDK Instructions

**NET\_SDK\_CloseAlarmChan**

cancel channel of uploading alarm.

BOOL NET\_SDK\_CloseAlarmChan( LONG *lAlarmHandle*

);

**Parameters**

*lAlarmHandle*

[in] return value of NET\_SDK\_SetupAlarmChan()

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_SetupAlarmChan](#_page_315_0)

Client SDK Instructions

**NET\_SDK\_PTZControl**

PTZ control operation(need to start previewing image)

BOOL NET\_SDK\_PTZControl( LONG *lLiveHandle*, DWORD *dwPTZCommand*, DWORD *dwSpeed*

);

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *dwPTZCommand*

[in] PTZ control command,list as follows:

**macro definition**

PTZ\_CMD\_STOP

PTZ\_CMD\_LEFT

PTZ\_CMD\_RIGHT

PTZ\_CMD\_UP

PTZ\_CMD\_DOWN

PTZ\_CMD\_LEFT\_UP

**value of maro definition**

0

1

2

3

4

5

**meaning**

PTZ is stopped

PTZ turns left

PTZ turns right

PTZ turns pitch up

PTZ turns pitch under

PTZ turns upleft

PTZ\_CMD\_LEFT\_DOWN 6

PTZ\_CMD\_RIGHT\_UP 7

PTZ\_CMD\_RIGHT\_DOWN 8

PTZ\_CMD\_NEAR 9

PTZ\_CMD\_FAR 10

PTZ\_CMD\_ZOOM\_OUT 11

PTZ\_CMD\_ZOOM\_IN 12

PTZ\_CMD\_IRIS\_OPEN 13

PTZ\_CMD\_IRIS\_CLOSE 14

PTZ\_CMD\_RESET 0xF0

PTZ turns downleft

PTZ turns upright

PTZ turns downright

adjust focus fore

adjust focus aft

focus length decreases

focus length increases

open aperture

close aperture

reset PTZ state

*dwSpeed*

[in] speed of PTZ ,list as follows:

**macro definition**

PTZ\_SPEED\_1 PTZ\_SPEED\_2 PTZ\_SPEED\_3

PTZ\_SPEED\_4

**value of macro definition**

1 2 3

4

PTZ\_SPEED\_5 5 PTZ\_SPEED\_6 6 PTZ\_SPEED\_7 7 PTZ\_SPEED\_8 8

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each command operated on PTZ needs to call this interface twice,start command and stop command,which command to call depends on the parameter *dwPTZCommand*. Before calling this interface preview should be open.Each operation on PTZ corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

PTZ default speed is the top speed.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

**Android Interface**

The corresponding Android interface

boolean long int int

);

PTZControl( handle, command, speed

Client SDK Instructions

**NET\_SDK\_PTZControl\_Other**

PTZ control operation(no need to start image preview)

BOOL NET\_SDK\_PTZControl\_Other( LONG *lUserID*,

LONG *lChannel*, DWORD *dwPTZCommand*, DWORD *dwSpeed*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0 *dwPTZCommand*

[in] PTZ control command,list as follows:

**macro definition**

PTZ\_CMD\_STOP

PTZ\_CMD\_LEFT

PTZ\_CMD\_RIGHT

PTZ\_CMD\_UP

PTZ\_CMD\_DOWN

**value of macro definition**

0

1

2

3

4

**meaning**

PTZ is stopped

PTZ turns left

PTZ turns right

PTZ turns pitch up

PTZ turns pitch under

PTZ\_CMD\_LEFT\_UP 5

PTZ\_CMD\_LEFT\_DOWN 6

PTZ\_CMD\_RIGHT\_UP 7

PTZ\_CMD\_RIGHT\_DOWN 8

PTZ\_CMD\_NEAR 9

PTZ\_CMD\_FAR 10

PTZ\_CMD\_ZOOM\_OUT 11

PTZ\_CMD\_ZOOM\_IN 12

PTZ\_CMD\_IRIS\_OPEN 13

PTZ\_CMD\_IRIS\_CLOSE 14

PTZ\_CMD\_RESET 0xF0

PTZ turns upleft

PTZ turns downleft

PTZ turns upright

PTZ turns downright

adjust focus fore

adjust focus aft

focus length decreases

focus length increases

open aperture

close aperture

reset PTZ state

*dwSpeed*

[in] speed of PTZ ,list as follows:

**macro definition**

PTZ\_SPEED\_1

PTZ\_SPEED\_2

**value of macro definition**

1

2

PTZ\_SPEED\_3 3 PTZ\_SPEED\_4 4 PTZ\_SPEED\_5 5 PTZ\_SPEED\_6 6 PTZ\_SPEED\_7 7 PTZ\_SPEED\_8 8

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each command operated on PTZ needs to call this interface twice,start command and stop command,which command to call depends on the parameter *dwPTZCommand*. Before calling this interface preview should be open.Each operation on PTZ corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

PTZ default speed is the top speed.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

**Android Interface**

The corresponding Android interface

boolean long long int int

);

PTZControl\_Other( userId,

channel, command, speed

Client SDK Instructions

**NET\_SDK\_PTZPreset**

PTZ preset point operation(need to open preview)

BOOL NET\_SDK\_PTZPreset( LONG *lLiveHandle*, DWORD *dwPTZPresetCmd*, DWORD *dwPresetIndex* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *dwPTZPresetCmd*

[in] operate PTZ preset point command,list as follows:

**macro definition**

PTZ\_CMD\_PRESET\_SET

PTZ\_CMD\_PRESET\_GO

PTZ\_CMD\_PRESET\_DEL

**value of macro definition**

16

17

18

**meaning**

PTZ sets preset points

to appointed preset point

delete preset points

*dwPresetIndex*

[in] serial number of preset point,at most support 255 points

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_PTZPreset\_Other**

PTZ preset point operation

BOOL NET\_SDK\_PTZPreset\_Other( LONG *lUserID*,

LONG *lChannel*,

DWORD *dwPTZPresetCmd*, DWORD *dwPresetIndex* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0 *dwPTZPresetCmd*

[in] operate PTZ preset point command,list as follows:

**macro definition**

PTZ\_CMD\_PRESET\_SET

PTZ\_CMD\_PRESET\_GO

PTZ\_CMD\_PRESET\_DEL

**value of macro definition**

16

17

18

**meaning**

PTZ sets preset points

to appointed preset point

delete preset points

*dwPresetIndex*

[in] serial number of preset point,at most support 255 points

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_PTZSetCruise**

set PTZ cruise line

BOOL NET\_SDK\_PTZSetCruise(

LONG *lLiveHandle*, BYTE *byCruiseRoute*, [DD\_CRUISE\_POINT\_INFO](#_page_549_0) *\*pCruisePoint*, WORD *pointNum*

);

**Parameters**

*lLiveHandle*

[in] play handle *byCruiseRoute*

[in] cruise line *\*pCruisePoint*

[in] cruise point *pointNum*

[in] count of cruise point

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZSetCruise\_Other](#_page_337_0)

Client SDK Instructions

**NET\_SDK\_PTZSetCruise\_Other**

set PTZ cruise line operation

BOOL NET\_SDK\_PTZSetCruise\_Other( LONG *lUserID*, LONG *lChannel*,

BYTE *byCruiseRoute*, [DD\_CRUISE\_POINT\_INFO](#_page_549_0) *\*pCruisePoint*, WORD *pointNum*

);

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number,start from 0 *byCruiseRoute*

[in] cruise line *\*pCruisePoint*

[in] cruise point *pointNum*

[in] count of cruise point

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZSetCruise](#_page_335_0)

Client SDK Instructions

**NET\_SDK\_PTZCruise**

PTZ cruise operation,need to start preview

BOOL NET\_SDK\_PTZCruise( LONG *lLiveHandle*, DWORD *dwPTZCruiseCmd*, BYTE *byCruiseRoute*, );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *dwPTZCruiseCmd*

[in] operate PTZ cruise command,list as follows:

**value of**

**macro definition** **macro** **meaning definition**

PTZ\_CMD\_CRUISE\_CFG 19

set cruise line,amount to execute Enter,Set and Leave commands

enter cruise mode and then cruise

PTZ\_CMD\_ENTER\_CURISE\_MODE20 preset point setting is permitted

PTZ\_CMD\_LEAVE\_CURISE\_MODE 22

quit cruise setting

PTZ\_CMD\_CRUISE\_RUN 23

PTZ\_CMD\_CRUISE\_STOP 24

PTZ\_CMD\_CRUISE\_DEL 25

choose a cruise line to cruise

PTZ stops cruise

delete cruise line

*byCruiseRoute*

[in] cruise path,at most support 32 pathes

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_PTZCruise\_Other**

PTZ cruise operation

BOOL NET\_SDK\_PTZCruise\_Other( LONG *lUserID*,

LONG *lChannel*,

DWORD *dwPTZCruiseCmd*, BYTE *byCruiseRoute*, );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,start from 0 *dwPTZCruiseCmd*

[in] operate PTZ cruise command,list as follows:

**value of**

**macro definition** **macro** **meaning definition**

PTZ\_CMD\_CRUISE\_CFG 19

set cruise line,amount to execute Enter,Set and Leave commands

PTZ\_CMD\_ENTER\_CURISE\_MODE20 enter cruise mode and then cruise preset point

PTZ\_CMD\_LEAVE\_CURISE\_MODE 22

setting is permitted

quit cruise setting

PTZ\_CMD\_CRUISE\_RUN 23

PTZ\_CMD\_CRUISE\_STOP 24

PTZ\_CMD\_CRUISE\_DEL 25

choose a cruise line to cruise

PTZ stops cruise

delete cruise line

*byCruiseRoute*

[in] cruise path,at most support 32 pathes

**Return Values**

TRUE means success; FALSE means failure. to get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_PTZTrack**

PTZ track operation,need to open preview

BOOL NET\_SDK\_PTZTrack( LONG *lLiveHandle*, DWORD *dwPTZTrackCmd* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *dwPTZTrackCmd*

[in] operate PTZ track command,list as follows:

**value of**

**macro definition** **macro meaning definition**

PTZ\_CMD\_TRACK\_START 26

PTZ\_CMD\_TRACK\_STOP 27

start track

stop track

start to PTZ\_CMD\_TRACK\_START\_RECORD28 store

track

stop PTZ\_CMD\_TRACK\_STOP\_RECORD 29 storing

track

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_DVR\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_LivePlay](#_page_156_0)

Client SDK Instructions

**NET\_SDK\_PTZTrack\_Other**

PTZ track operation

BOOL NET\_SDK\_PTZTrack\_Other( LONG *lUserID*,

LONG *lChannel*, DWORD *dwPTZTrackCmd* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,starting with 0 *dwPTZTrackCmd*

[in] operate PTZ track command,list as follows:

**macro definition**

PTZ\_CMD\_TRACK\_START PTZ\_CMD\_TRACK\_STOP

PTZ\_CMD\_TRACK\_SET

PTZ\_CMD\_TRACK\_DEL

**value of macro definition**

26 27

28

29

**meaning**

start track stop track

start to store track

stop storing track

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_DVR\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_PTZAutoScan**

PTZ automatic scan operation

BOOL NET\_SDK\_PTZAutoScan( LONG *lLiveHandle*, DWORD *dwPTZAutoScanCmd*, DWORD *dwSpeed*,

BOOL *bIsAutoScan* );

**Parameters**

*lLiveHandle*

[in] play handle *dwPTZAutoScanCmd*

[in] PTZ automatic scan command, list as follows:

**value of**

**macro definition** **macro meaning definition**

start PTZ\_CMD\_AUTO\_SCAN\_START29 automatic

scan

stop PTZ\_CMD\_AUTO\_SCAN\_STOP 30 automatic

scan

*dwSpeed*

[in] speed of PTZ ,list as follows:

**macro definition**

**value of macro definition**

PTZ\_SPEED\_1 1 PTZ\_SPEED\_2 2 PTZ\_SPEED\_3 3 PTZ\_SPEED\_4 4 PTZ\_SPEED\_5 5 PTZ\_SPEED\_6 6 PTZ\_SPEED\_7 7 PTZ\_SPEED\_8 8

*bIsAutoScan*

[in] Scan mode, TURE means automatic scan, and FALSE represents random scan

**Return Values**

TURE means success,FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZAutoScan\_Other](#_page_354_0)

Client SDK Instructions

**NET\_SDK\_PTZAutoScan\_Other**

PTZ automatic scan operation

BOOL NET\_SDK\_PTZAutoScan\_Other( LONG *lUserID*,

LONG *lChannel*,

DWORD *dwPTZAutoScanCmd* );

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number,starts from 0 *dwPTZAutoScanCmd*

[in] PTZ automatic scan command, list as follows:

**value of**

**macro definition** **macro meaning definition**

start PTZ\_CMD\_AUTO\_SCAN\_START30 automatic

scan

stop PTZ\_CMD\_AUTO\_SCAN\_STOP 31 automatic

scan

***dwSpeed***

**[in] PTZ speed**

**Macro Definition** **Meaning**

PTZ\_SPEED\_1 1 PTZ\_SPEED\_2 2 PTZ\_SPEED\_3 3 PTZ\_SPEED\_4 4 PTZ\_SPEED\_5 5 PTZ\_SPEED\_6 6 PTZ\_SPEED\_7 7 PTZ\_SPEED\_8 8

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZAutoScan](#_page_351_0)

**Android Interface**

The corresponding Android interface

boolean long long int

);

PTZAutoScanOther( lUserID, lChannel, dwPTZAutoScanCmd

Client SDK Instructions

**NET\_SDK\_PTZControl\_3D**

PTZ 3D control operation(need to start previewing image)

BOOL NET\_SDK\_PTZControl\_3D( LONG *lLiveHandle*, LONG *lChannel*,

[PTZ\_3D\_POINT\_INFO](#_page_710_0) *\*pPtz3DInfo* );

**Parameters**

*lLiveHandle*

[in] return value of NET\_SDK\_LivePlay *lChannel*

[in] channel number,starting with 0 *\*pPtz3DInfo*

[in] information about PTZ 3D control,as follows **PTZ\_3D\_POINT\_INFO**

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZControl](#_page_319_0)

Client SDK Instructions

**NET\_SDK\_PTZControl\_3D\_Ex**

PTZ 3D control operation(no need to start previewing image)

BOOL NET\_SDK\_PTZControl\_3D\_Ex( LONG *lUserID*,

LONG *lChannel*, [PTZ\_3D\_POINT\_INFO](#_page_710_0) *\*pPtz3DInfo* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *lChannel*

[in] channel number,starting with 0 *\*pPtz3DInfo*

[in] information about PTZ 3D control,as follows **PTZ\_3D\_POINT\_INFO**

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_PTZControl](#_page_319_0)

Client SDK Instructions

**NET\_SDK\_GetPTZCameraType**

get camera type(only support IPC)

BOOL NET\_SDK\_PTZPreset\_Other( LONG *lUserID*,

LONG *NET\_SDK\_CAMERA\_TYPE \*pCameraType* );

**Parameters**

*lUserID*

[in] return value of userID *NET\_SDK\_CAMERA\_TYPE \*pCameraType*

[in] reture value of camera type,list as follows:

**value of**

**camera type** **camera meaning type**

NET\_SDK\_NOT\_SUPPORT\_PTZ 0

NET\_SDK\_DOME\_SUPPORT\_PTZ 1

NET\_SDK\_SUPPORT\_PTZ 2

NET\_SDK\_PTZ\_END

bullet camera don't support ptz

bullet camera support ptz

dome camera support ptz

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Each operation on PTZ is corresponding to each control code between device and PTZ,device sends control code to PTZ according to current decoder type and decoder address. If current decoder doesnot match PTZ ,reconfig decoder is necessary.

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**typedef enum \_\_Channel\_type\_\_ {**

**E\_NULL\_CHL\_TYPE, E\_DIGITAL\_CHL\_TYPE,** **//Digital**

**Channel E\_ANALOG\_CHL\_TYPE, E\_ALARMOUT\_CHL\_TYPE,**

**Output Channel E\_SENSOR\_CHL\_TYPE,**

**//Analog Channel //Alarm**

**//Sensor Channel**

**}CHANNEL\_TYPE;**

**typedef struct \_net\_sdk\_channel\_ptz {**

**unsigned int** **dwChannel;//Channel No.(starting with zero)**

**CHANNEL\_TYPE Type**

**unsigned char }NET\_SDK\_CHANNEL\_PTZ;**

**eChanneltype;//Channel**

**resv[8];**

**NET\_SDK\_GetSupportPtzList**

Get the channel of NVR in favor of PTZ list（only for N9000 device）

BOOL NET\_SDK\_GetSupportPtzList( LONG lUserID,

int listNum,

NET\_SDK\_CHANNEL\_PTZ \*pOutChannelPtz, int \*returnListNum,

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *listNum*

[in] the number of NET\_SDK\_CHANNEL\_PTZ supported by the request memory of pOutChannelPtz

*\*pOutChannelPtz*

[out] Return PTZ information list *\*returnListNum*

[out] the effective number of returning PTZ information list

Return Values

TRUE means success; FALSE means failure. To get error information, call [NET\_SDK\_GetLastError](file:///C:/Users/Administrator/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C3_43120.7263751968/interface/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_GetPTZConfig**

Get the related configuration of PTZ (only for N9000).

BOOL NET\_SDK\_GetPTZConfig( LONG *lUserID*,

LONG *lChannel*, DWORD *dwCommand*, LPVOID *lpInBuffer*, DWORD *dwInBufferSize*, LPVOID *lpOutBuffer*,

DWORD *dwOutBufferSize*, LPDWORD *lpBytesReturned* );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *lChannel*

[in] channel number starts from 0. -1 means all channels.If the command don't need channel number, this parameter is invalid.

*dwCommand*

[in] device configuration command, see Configuration Commands *lpInBuffer*

[in] buffer pointer to sending data. it is can not be NULL . *dwInBufferSize*

[in] the buffer length of sending data (in byte). It is can not be 0. *lpOutBuffer*

[out] the buffer pointer to receiving data. *dwOutBufferSize*

[in] the buffer length of receiving data (in byte). It is can not be 0 *lpBytesReturned*

[out] the length pointer to the data actually received. it is can not be null

**Remarks**

Different functions has different structures and commands as shown below.

**dwCommand**

**Macro Definition**

**dwCommand**

**Meanings**

**Sending Structure**

**Receiving Structu**

DD\_PTZ\_CONFIG\_PRESET

DD\_PTZ\_CONFIG\_CRUISE

Get preset NULL

Get cruise NULL

DD\_PTZ\_PRESET\_CON

DD\_CH\_CRUISE

DD\_PTZ\_CONFIG\_CRUISE\_POINTthe curise ts f int(cruiseIndex)DD\_CRUISE\_POINT\_IN

**See Also**

[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_StartVoiceCom**

start talkback

LONG NET\_SDK\_StartVoiceCom( LONG *lUserID*,

BOOL *bNeedCBNoEncData*, [TALK\_DATA\_CALLBACK](#_page_376_0) *fVoiceDataCallBack*, void *\* pUser*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *bNeedCBNoEncData*

[in] whether need to encode voice when local voice data callback. If FALSE is selected, the stream from the device will be decoded first and then it is called back.

*fVoiceDataCallBack*

[in] function of audio data callback *PUser*

[in] user data

**Return Values**

-1 means failure and other value is handle parameter of NET\_SDK\_StopVoiceCom. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StopVoiceCom](#_page_380_0) [NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_SetVoiceComClientVolume**

set client volume of talkback

BOOL NET\_SDK\_SetVoiceComClientVolume( LONG *lVoiceComHandle*,

WORD *wVolume* );

**Parameters**

*lVoiceComHandle*

[in] return value of NET\_SDK\_StartVoiceCom *wVolume*

[in] set volume,range[0,0xffff]

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StartVoiceCom](#_page_372_0)

Client SDK Instructions

**TALK\_DATA\_CALLBACK**

data callback in talkback

void \*TALK\_DATA\_CALLBACK(

LONG *lVoiceComHandle*, char *\*pRecvDataBuffer*, DWORD *dwBufSize*,

BYTE *byAudioFlag*, void *\* pUser*

);

**Parameters**

*lVoiceComHandle*

[in] interface handle of talkback *\*pRecvDataBuffer*

[in] buffer pointer to receive talkback data,received talkback data is PCM data without encoding

*dwBufSize*

[in] size of buffer area *byAudioFlag*

[in] audio mark,the only value is 1,and it means the talkback data from device

*pUser*

[in] pointer to user information

**Return Values**

None. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_GetAudioInfo**

Get audio information.

BOOL NET\_SDK\_GetAudioInfo( LONG *lVoiceComHandle*, void *\*pAudioInfo*,

LONG *infoLen* );

**Parameters**

*lVoiceComHandle*

[in] the return value of NET\_SDK\_StartVoiceCom() *pAudioInfo*

[in] a pointer to audio information *infoLen*

[in] the length of audio information

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_ReleaseAudioEncoder](#_page_392_0) [NET\_SDK\_EncodeAudi](#_page_387_0)

[oFrame](#_page_387_0)

>

Client SDK Instructions

**NET\_SDK\_StopVoiceCom**

stop talkback or voice forward

BOOL NET\_SDK\_StopVoiceCom( LONG *lVoiceComHandle* );

**Parameters**

*lVoiceComHandle*

[in] return value of NET\_SDK\_StartVoiceCom or NET\_SDK\_StartVoiceCom\_MR

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StartVoiceCom](#_page_372_0) [NET\_SDK\_StartVoiceCom\_MR](#_page_382_0)

Client SDK Instructions

**NET\_SDK\_StartVoiceCom\_MR**

start voice forward function

LONG NET\_SDK\_StartVoiceCom\_MR( LONG *lUserID*,

[TALK\_DATA\_CALLBACK](#_page_376_0) *fVoiceDataCallBack*, void *\* pUser*

);

**Parameters**

*lUserID*

[in] user ID *bNeedNoEncodeData*

[in] whether to encode the language data. If TRUE is selected, the stream from the device will be directly called back and the stream transfered to the device is encoded by SDK . If FALSE is selected, the stream from the deivce is decoded first and then called back; the stream transfered to the device is not encoded by SDK.

*fVoiceDataCallBack*

[in] return value of voice data callback function *pUser*

[in] user information

**Return Values**

-1 means failure and other value is return value of voice forward. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_VoiceComSendData](#_page_385_0)

Client SDK Instructions

**NET\_SDK\_VoiceComSendData**

send data when voice forward

BOOL NET\_SDK\_VoiceComSendData( LONG *lVoiceComHandle*, char *\*pSendBuf*,

DWORD *dwBufSize* );

**Parameters**

*lVoiceComHandle*

[in] handle of voice component *pSendBuf*

[in] buffer pointer to send data, the talkback data in buffer area is PCM data without encoding

*dwBufSize*

[in] size of buffer area

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_StartVoiceCom\_MR](#_page_382_0)

Client SDK Instructions

**NET\_SDK\_EncodeAudioFrame**

encode audio

BOOL NET\_SDK\_EncodeAudioFrame( LONG *lEncodeHandle*, unsigned char *\*pInBuffer*, LONG *inLen*, unsigned char *\* pOutBuffer* , LONG *\*pOutLen*

);

**Parameters**

*IEncodeHandle*

[in] handle of encoding audio, the return value of NET\_SDK\_InitAudioEncoder()

*\*pInBuffer*

[in] buffer area for input,get PCM audio data according to sample standard(sample frequency is 16000,16 bytes, signal channel),the standard size of input data is 1280 bytes.

*inLen*

[out] buffer area for output,data length after encoding *pOutBuffer*

[out] buffer area for output,the output data size after encoding is 80 bytes.

*\*pOutLen*

[out] buffer area for output,output data length after encoding

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

It's set to mate with talkback & forward function.When client sends original audio data to device, firstly compress and encode audio through function of audio encoding,and then send to device. Client gets the compressed code stream,and then call NET\_SDK\_DecodeAudioFrame to decode data. Before calling encode and decode function initialization is needed, also when call function is finished,free resource.

**See Also**

[NET\_SDK\_InitAudioEncoder](#_page_390_0)

Client SDK Instructions

**NET\_SDK\_InitAudioEncoder**

initialize audio encoder

void\* NET\_SDK\_InitAudioEncoder( void *\*pAudioInfo*,

LONG *infoLen* );

**Parameters**

*pAudioInfo*

[in] pointer to audio information *infoLen*

[in] length of audio information

**Return Values**

-1 means failure and other value is handle of audio encoding. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_ReleaseAudioEncoder](#_page_392_0) [NET\_SDK\_EncodeAudioFr](#_page_387_0)

[ame](#_page_387_0)

Client SDK Instructions

**NET\_SDK\_ReleaseAudioEncoder**

free audio encoding resource

void NET\_SDK\_ReleaseAudioEncoder( LONG *\*IEncodeHandle*

);

**Parameters**

*IEncodeHandle*

[in] handle of audio encoding,return value of NET\_SDK\_InitAudioEncoder

**Return Values**

None. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_InitAudioEncoder](#_page_390_0)

Client SDK Instructions

**NET\_SDK\_DecodeAudioFrame**

audio decode

BOOL NET\_SDK\_DecodeAudioFrame( LONG *lDecodeHandle*, unsigned char *\*pInBuffer*, LONG *inLen*, unsigned char *\* pOutBuffer*, LONG *\*pOutLen*

);

**Parameters**

*IDecodeHandle*

[in] audio encode handle,return value of NET\_SDK\_InitAudioDecoder()

*\*pInBuffer*

[in] input buffer area,get PCM audio data according to sample standard(sampling frequency is in000,16 bits, signal channel),standard data size of input is 1280 bytes

*inLen*

[out] output buffer are,data length after encoding *pOutBuffer*

[out] output buffer area,output data size after encoding is 80 bytes

*\*pOutLen*

[out] output buffer area,output data length after encoding

**Return Values**

TRUE means success and FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

It's set to mate with talkback & forward function.When client send original audio data to device, firstly compress and encode audio through function of audio encoding,and then send to device. Client gets the compressed code stream,and then call this interface to decode data. Before calling encode and decode function initialization is needed, also when call function is finished,free resource.

**See Also**

[NET\_SDK\_InitAudioDecoder](#_page_397_0)

Client SDK Instructions

**NET\_SDK\_InitAudioDecoder**

initialize audio decoder

LONG NET\_SDK\_InitAudioDecoder( void *\*pAudioInfo*,

LONG *infoLen* );

**Parameters**

*pAudioInfo*

[in] pointer to audio information *infoLen*

[in] length of audio information

**Return Values**

-1 means failure and other value is handle of audio decoding. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_ReleaseAudioDecoder](#_page_399_0) [NET\_SDK\_DecodeAudioFr](#_page_394_0)

[ame](#_page_394_0)

Client SDK Instructions

**NET\_SDK\_ReleaseAudioDecoder**

free audio decoding resource

void NET\_SDK\_ReleaseAudioDecoder( LONG *IDecodeHandle*

);

**Parameters**

*IDecodeHandle*

[in] handle of audio decoding,return value of NET\_SDK\_InitAudioDecoder

**Return Values**

None. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_InitAudioDecoder](#_page_397_0)

Client SDK Instructions

**NET\_SDK\_FormatDisk**

format harddisk remotely(only support 3.0DVR)

LONG NET\_SDK\_FormatDisk( LONG *lUserID*,

LONG *lDiskNumber* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *lDiskNumber*

[in] harddisk number,start from 0,0xff is valid to all harddisks(except read-only harddisk)

**Return Values**

-1 means failure and other value is parameter of function NET\_SDK\_CloseFormatHandle. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

If network breaks in formation process,formation on device continues but client can't receive the state.

**See Also**

[NET\_SDK\_CloseFormatHandle](#_page_406_0) [NET\_SDK\_GetFormatProgre](#_page_403_0)

[ss](#_page_403_0) [NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetFormatProgress**

get progress of formatting harddisk(only support 3.0DVR)

BOOL NET\_SDK\_GetFormatProgress( LONG *lFormatHandle*,

LONG *\*pCurrentFormatDisk*, LONG *\*pCurrentDiskPos*, LONG *\*pFormatStatic*

);

**Parameters**

*lFormatHandle*

[in] handle of save formatting harddisk,return value of NET\_SDK\_FormatDisk

*pCurrentFormatDisk*

[out] pointer to save current formatting harddisk number,harddisk number starts from 0,-1 is the initial value

*pCurrentDiskPos*

[out] pointer to save current progress of formatting harddisk,progress range is 0-100

*pFormatStatic*

[out] pointer to save the state of formatting harddisk,0-formatting;1-finish formation; 2-formatting current harddisk makes mistake,formation can't continue,this error shows both in local and network harddisks; 3-formation of current harddisk can't start due to lost of network harddisk caused by network exception

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FormatDisk](#_page_401_0)

Client SDK Instructions

**NET\_SDK\_CloseFormatHandle**

close handle of formatting harddisk,and free resource(only support 3.0DVR)

BOOL NET\_SDK\_CloseFormatHandle( LONG *lFormatHandle*

);

**Parameters**

*lFormatHandle*

[in] return value of NET\_SDK\_ FormatDisk()

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FormatDisk](#_page_401_0)

Client SDK Instructions

**NET\_SDK\_FindDisk**

Get the HDD information of the device.

POINTERHANDLE NET\_SDK\_FindDisk( LONG *lUserID*,

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login()

**Return Values**

Return to the handle of getting HDD information. The value is greater than 0, which means

success. [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

NET\_SDK\_GetNextDiskInfo NET\_SDK\_FindDiskClose [NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetNextDiskInfo**

Get the information of the HDD in order (call up serveral times until return failure).

BOOL NET\_SDK\_GetNextDiskInfo(

POINTERHANDLE *lDiskHandle* , NET\_SDK\_DISK\_INFO *\*pDiskInfo*,

);

**Parameters**

*lDiskHandle*

[in] the return value of NET\_SDK\_FindDisk *pDiskInfo*

[out] return to the HDD information

**Return Values**

The return value is greater than 0 which means success. [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

NET\_SDK\_FindDisk NET\_SDK\_FindDiskClose [NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_FindDiskClose**

Having finished getting the HDD information of the device, release resources.

BOOL NET\_SDK\_FindDiskClose( POINTERHANDLE lDiskHandle, );

**Parameters**

*lDiskHandle*

[in] the return value of NET\_SDK\_FindDisk

**Return Values**

The value is greater than 0, which means success. [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

NET\_SDK\_FindDisk NET\_SDK\_GetNextDiskInfo

Client SDK Instructions

**NET\_SDK\_ActiveDevice**

active the IPC device

BOOL NET\_SDK\_ActiveDevice( char *\*pIp,*

int iPort,

char *\*password* );

**Parameters**

*pIp*

[in] the IP address of the device to be activated(can be the second ip starting with 169.254)

*iPort*

[in] the http port of the device is to be activated *password*

[in] the password to activate the device

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_GetUpgradeState](file:///C:/Users/A10609/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_44538.6175127778/interface/NET_SDK_GetUpgradeState.htm) [NET\_SDK\_GetUpgradePro](file:///C:/Users/A10609/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_44538.6175127778/interface/NET_SDK_GetUpgradeProgress.htm) [gres](file:///C:/Users/A10609/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_44538.6175127778/interface/NET_SDK_GetUpgradeProgress.htm)

Client SDK Instructions

**NET\_SDK\_Upgrade**

remote upgrade

LONG NET\_SDK\_Upgrade( LONG *lUserID*, char *\*sFileName*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *wVolume*

[in] directory of file upgrade(include file name)

**Return Values**

-1 means failure and other values are parameter of NET\_SDK\_GetUpgradeState. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0) [NET\_SDK\_CloseUpgradeHandle](#_page_423_0)

[NET\_SDK\_GetUpgradeState](#_page_419_0) [NET\_SDK\_GetUpgradeProgres](#_page_421_0)

Client SDK Instructions

**NET\_SDK\_GetUpgradeState**

get state of remote upgrading

int NET\_SDK\_GetUpgradeState( LONG *lUpgradeHandle*

);

**Parameters**

*lUpgradeHandle*

[in] return value of NET\_SDK\_Upgrade

**Return Values**

-1 means failure. Other values' meanings are as follows: 1-finish upgrading

2-upgrading 3-failed

4-network breaks,unknown state

5-language version of upgrading file doesnot match

Get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Upgrade](#_page_417_0)

Client SDK Instructions

**NET\_SDK\_GetUpgradeProgress**

get progress of remote upgrading

int NET\_SDK\_GetUpgradeProgress( LONG *lUpgradeHandle*

);

**Parameters**

*lUpgradeHandle*

[in] return value of NET\_SDK\_Upgrade

**Return Values**

-1 means failure and 0-100 means progress of upgrading. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Upgrade](#_page_417_0)

Client SDK Instructions

**NET\_SDK\_CloseUpgradeHandle**

close handle of remote upgrade,and free resource

BOOL NET\_SDK\_CloseUpgradeHandle( LONG *lUpgradeHandle*

);

**Parameters**

*lUpgradeHandle*

[in] return value of NET\_SDK\_Upgrade()

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Upgrade](#_page_417_0)

Client SDK Instructions

**NET\_SDK\_UpgradeIPC**

remote upgrade IPC

LONG NET\_SDK\_UpgradeIPC( LONG *lUserID*,

char *\*sFileName,* unsigned int *fileType* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *sFileName*

[in] directory of file upgrade(include file name) *fileType*

[in] type of the upgrade

file; 0:software;1:kernel;2:Uboot;3:AIlib

**Return Values**

-1 means failure and other values are parameter of NET\_SDK\_GetUpgradeState. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0) [NET\_SDK\_CloseUpgradeHandle](#_page_423_0)

[NET\_SDK\_GetUpgradeState](#_page_419_0) [NET\_SDK\_GetUpgradeProgres](#_page_421_0)

Client SDK Instructions

**NET\_SDK\_FindDVRLog**

find log information of device(not support IPC)

LONG NET\_SDK\_FindDVRLog( LONG *lUserID*, DWORD *dwType*,

[DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *dwType*

[in] type of log(examples see N9000\_LOG\_MAJOR\_TYPE )

*\*lpStartTime*

[in] the start time of file *\*lpStopTime*

[in] the stop time of file

**Return Values**

-1 means failure and other values are part of parameters in function NET\_DVR\_FindNextLog. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

This interface is used to search normal log information,and the maximum capacity is 2000.

**See Also**

[NET\_SDK\_Login](#_page_138_0) [NET\_SDK\_FindNextLog](#_page_430_0) [NET\_SDK\_FindLog](#_page_432_0)

[Close](#_page_432_0)

Client SDK Instructions

**NET\_SDK\_FindNextLog**

get log information one by one(not support IPC)

LONG NET\_SDK\_FindNextLog(

LONG *lLogHandle*, [LPNET\_SDK\_LOG](#_page_694_0) *lpLogData* );

**Parameters**

*lLogHandle*

[in] handle for finding log.return value of NET\_SDK\_FindDVRLog()

*lpLogData*

[out] pointer to store log information

**Return Values**

-1 means failure and other values are current state information. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**

Before calling this interface,call [NET\_SDK\_FindDVRLog](#_page_427_0) to get current handle for finding.

**See Also**

[NET\_SDK\_FindDVRLog](#_page_427_0)

Client SDK Instructions

**NET\_SDK\_FindLogClose**

free resource of finding log(not support IPC)

BOOL NET\_SDK\_FindLogClose( LONG *lLogHandle*

);

**Parameters**

*lLogHandle*

[in] handle of finding log,return value of NET\_SDK\_FindDVRLog()

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_FindDVRLog](#_page_427_0)

Client SDK Instructions

**NET\_SDK\_FindEventInfo**

find event info(only support 3.0DVR)

LONG NET\_SDK\_FindEvent( LONG *lUserID*, DWORD *dwType*

ULONGLONG *channlMask*, [DD\_TIME](#_page_635_0) *\*lpStartTime*, [DD\_TIME](#_page_635_0) *\*lpStopTime* );

**Parameters**

*lUserID*

[in] returned value of NET\_SDK\_Login() *dwType*

[in] event type,refer to DD\_EVENT\_TYPE:

**Type** **Value**

DD\_EVENT\_TYPE\_MOTION 0x0001 DD\_EVENT\_TYPE\_SENSOR 0x0002 DD\_EVENT\_TYPE\_V\_LOSS 0x0004 DD\_EVENT\_TYPE\_V\_COVER 0x0008

*channlMask*

[in] event happend in which channel, ((ULONGLONG)0x1 << N) N is search channel.

*\*lpStartTime*

[in] event starting time *\*lpStopTime*

[in] event ending time

**Return Values**

-1 means failure and other values are parameter of NET\_SDK\_FindNextEventInfo. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_FindNextEventInfo](#_page_437_0) [NET\_SDK\_FindEventInfoClos](#_page_439_0)

[e](#_page_439_0)

Client SDK Instructions

**NET\_SDK\_FindNextEventInfo**

find event info one by one(only support 3.0DVR)

LONG NET\_SDK\_FindNextEventInfo(

LONG *lEventHandle*, [LPNET\_SDK\_EVENT](#_page_686_0) *lpEventData*

);

**Parameters**

*lEventHandle*

[in] handle of searching event info, the return value of NET\_SDK\_FindEventInfo()

*lpEventData*

[out] pointer of saving event info

**Return Values**

-1 means failure and other values are event info. To get error code, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

Before call this interface to search event info, please call NET\_SDK\_FindEventInfo() to get the search handle.

**See Also**

[NET\_SDK\_FindEventInfo](#_page_434_0) [NET\_SDK\_FindEventInfoClose](#_page_439_0)

Client SDK Instructions

**NET\_SDK\_FindEventInfoClose**

Close searching event info,free resource(only support 3.0DVR)

BOOL NET\_SDK\_FindEventInfoClose( LONG *lEventHandle*

);

**Parameters**

*lEventHandle*

[in] search handle

**Return Values**

TURE means success; FALSE means failure. To get error code,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_FindEventInfo](#_page_434_0) [NET\_SDK\_FindNextEventInfo](#_page_437_0)

Client SDK Instructions

**NET\_SDK\_GetDefaultVideoEffect**

get default video effect

BOOL NET\_SDK\_GetDefaultVideoEffect( LONG *lUserID*,

DWORD *\*pBrightValue*, DWORD *\*pContrastValue*, DWORD *\*pSaturationValue*, DWORD *\*pHueValue*

);

**Parameters**

*lUserID*

[in] user ID *\*pBrightValue*

[in] pointer to brightness value *\*pContrastValue*

[in] pointer to color contrast *\*pSaturationValue*

[in] pointer to color saturation *\*pHueValue*

[in] pointer to gray scale

**Return Values**

TURE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_GetVideoEffect](#_page_113_0) [NET\_SDK\_SaveVideoEffect](#_page_459_0) [NET](#_page_110_0)

[\_SDK\_SetVideoEffect](#_page_110_0)

Client SDK Instructions

**NET\_SDK\_SetConfigFile**

import configuration file

BOOL NET\_SDK\_SetConfigFile( LONG *lUserID*,

char *\*sFileName* );

**Parameters**

*lUserID*

[in] user ID,return value of NET\_SDK\_Login *sFileName*

[in] directory of saving configuration file(binary file)

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_DVR\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetConfigFile**

export configuration file

BOOL NET\_SDK\_GetConfigFile( LONG *lUserID*,

char *\*sFileName* );

**Parameters**

*lUserID*

[in] user ID,return value of NET\_SDK\_Login() *sFileName*

[in] directory of storing configuration file(binary file)

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetNvrRecordDays**

Query the number of days of video exists on the NVR device（works for NVRonly）

BOOL NET\_SDK\_GetNvrRecordDays( LONG *lUserID*, NET\_SDK\_NVR\_DISKREC\_DATE\_ITEM\*

pDiskRecDateInfo LONG lBuffSize, LONG\* pDISKCount

);

**Parameters**

*lUserID*

[in] returned value of NET\_SDK\_Login() *pDiskRecDateInfo*

[out] video days information structure pointer for the NVR device

*lBuffSize*

[in] size of NET\_SDK\_NVR\_DISKREC\_DATE\_ITEM structure

*pDISKCount*

[out] number of hard drives for the NVR

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

Each hard disk of the NVR device corresponds to a query result, with the video days format 2021-

11-27~2021-12-

07, indicating the existence of video on the hard d isk during this time period

Client SDK Instructions

**NET\_SDK\_ShutDownDVR**

close device

BOOL NET\_SDK\_ShutDownDVR( LONG *lUserID*

);

**Parameters**

*lUserID*

[in] user ID,return value of NET\_SDK\_Login

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_RebootDVR**

reboot device

BOOL NET\_SDK\_RebootDVR( LONG *lUserID*

);

**Parameters**

*lUserID*

[in] user ID,return value of NET\_SDK\_Login

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_ChangTime**

modify system time of device

BOOL NET\_SDK\_ChangTime( LONG *lUserID*, unsigned long *time* );

**Parameters**

*lUserID*

[in] user ID *time*

[in] device system time

**Return Values**

TURE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_FormatTime**

Transform inter time to format time

void NET\_SDK\_FormatTime( LONGLONG *intTime*, [DD\_TIME](#_page_635_0) *\*pFormatTime* );

**Parameters**

*intTime*

[in] inter time form NET\_SDK\_FRAME\_INFO, from 1970-01-01,00:00:00,unit is microsecond

*\*pFormatTime*

[in] DD\_TIME format time

**Return Values**

None.This interface transforms the *time* parameter of [NET\_SDK\_FRAME\_INFO](#_page_688_0) *to* [*DD\_TIMEf*](#_page_635_0)*ormat time. To get error code, please call* [*NET\_SDK\_GetLastError*](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SaveVideoEffect**

save setting of video effect

BOOL NET\_SDK\_SaveVideoEffect( LONG *lUserID*,

LONG *lChannel*, DWORD *dwBrightValue*,

DWORD *dwContrastValue*, DWORD *dwSaturationValue*, DWORD *dwHueValue*

);

**Parameters**

*lUserID*

[in] user ID *lChannel*

[in] channel number,start from 0 *dwBrightValue*

[in] value of brightness *dwContrastValue*

[in] value of constract *dwSaturationValue*

[in] value of saturation *dwHueValue*

[in] value of gray scale

**Return Values**

TURE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_GetVideoEffect](#_page_113_0) [NET\_SDK\_GetDefaultVideoEffect](#_page_441_0)

[NET\_SDK\_SetVideoEffect](#_page_110_0)

Client SDK Instructions

**NET\_SDK\_ModifyDeviceNetInfo**

Modify the network configuration of the device accoding to the matching MAC.

BOOL NET\_SDK\_ModifyDeviceNetInfo( NET\_SDK\_DEVICE\_IP\_INFO \*pDeviceIPInfo

);

**Parameters**

*pDeviceIPInfo*

[in] the network configuration of the device

**Return Values**

-1 means failure; other values means the returned information value. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

Client SDK Instructions

**NET\_SDK\_TransparentConfig**

Transparent API protocol Interface

BOOL NET\_SDK\_TransparentConfig( LONG *lUserID*,

char *\*sendXML*, char *\*strUrl*, LPVOID *lpOutBuffer*,

DWORD *dwOutBufferSize*, LPDWORD *lpBytesReturned* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login *sendXML*

[in] xml contents in API *strUrl*

[in] URL of API。（IP and port are not included. eg：the origninal URL of API protocol ishttp://[:port]/PtzStopCruise[/channelId]. Here the URL is PtzStopCruise/channelId） .

*lpOutBuffer*

[out] the buffer pointer of receiving data *dwOutBufferSize*

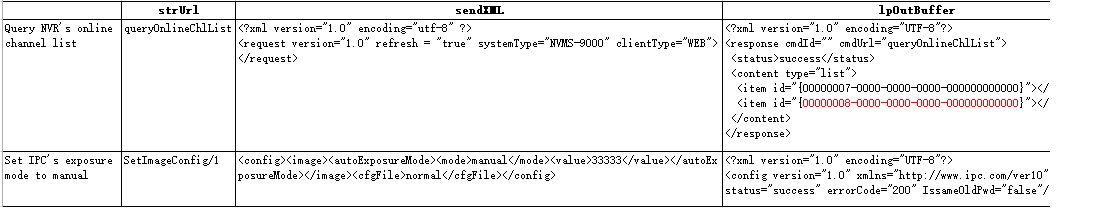
[in]the buffer size of receiving data in bytes can not be zero. *lpBytesReturned*

[out]a pointer to the data length actually received can not be null.

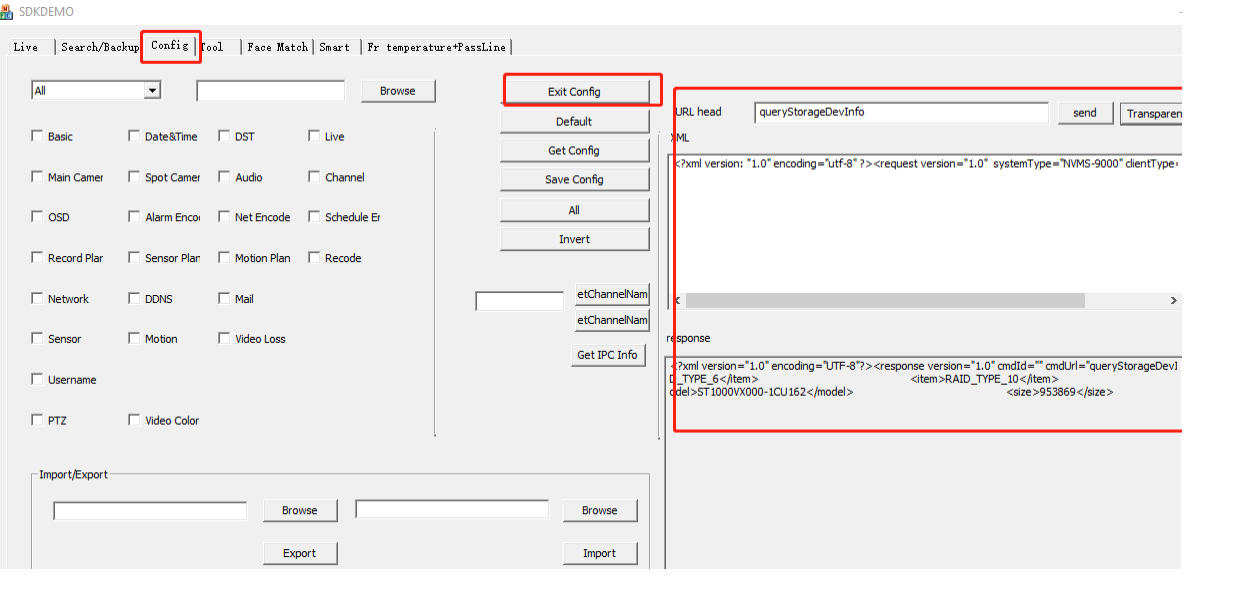
**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Remarks**



This function can be used to send http command to the device eg.

Test the function with the sdk demo: click the "Config" tab, input the strUrl and sendXML,click the "Transparent" button, the lpOutBuffer will display under

Client SDK Instructions

**NET\_SDK\_GetDeviceInfo**

Get parameters of decoding device

BOOL NET\_SDK\_GetDeviceInfo( LONG *lUserID*,

LPNET\_SDK\_DEVICEINFO *pdecviceInfo* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *pdecviceInfo*

[out] information about device parameter

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceTypeName**

get device type name

LONG NET\_SDK\_GetDeviceTypeName( LONG *lUserID*,

char *\*pNameBuffer*, long *bufferLen*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *\*pNameBuffer*

[out] type name buffer of device *bufferLen*

[out] buffer length of device type name

**Return Values**

The returned value is the device type name. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceTime**

get system time of device

BOOL NET\_SDK\_GetDeviceTime( LONG *lUserID*,

[DD\_TIME](#_page_635_0) *\*pTime* );

**Parameters**

*lUserID*

[in] user ID *\*pTime*

[in] pointer of device system time

**Return Values**

TURE means success,FALSE means failure.If live frame hasn't arrived client in 50 milliseconds, return value is False.Please try to call this interface more times until return value is True. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**Client SDK Instructions**

**NET\_SDK\_GetPTZCameraType**

Get camera type

BOOL NET\_SDK\_GetPTZCameraType( LONG *lUserID*, NET\_SDK\_CAMERA\_TYPE \*pCameraType *, );*

***Parameters***

*lUserID*

*[in] the return value of NET\_SDK\_Login() pCameraType*

*[in] Camera type as shown below:*

***Return Values***

*TRUE means sucess; FALSE means failure. To get error information, please call* [*NET\_SDK\_GetLastError*](mk:@MSITStore:C:UsersAdministratorDesktopDevice%20Net%20SDK%20manual.CHM::/NET_SDK_GetlastError.htm)

***Remarks***

*NET\_SDK\_CAMERA\_TYPE is an example. Please refer to the corresponding type of the libary function.*

***See Also***

[*NET SDK GetDVRConfig*](mk:@MSITStore:C:UsersAdministratorDesktopDevice%20Net%20SDK%20manual.CHM::/NET_SDK_LivePlay.htm)

***Camera***

***Camera type*** ***type Meanings value***

*The camera*

*NET\_SDK\_NOT\_SUPPORT\_PTZ 0* *don't support PTZ*

*NET\_SDK\_DOME\_SUPPORT\_PTZ1*

*The camera supports PTZ*

*NET\_SDK\_SUPPORT\_PTZ* *2*

*NET\_SDK\_PTZ\_END*

*The camera supports PTZ*

**Client SDK Instructions**

**NET\_SDK\_GetAlarmStatus**

Get the alarm information of the device.

BOOL NET\_SDK\_GetAlarmStatus( LONG lUserID,

LPVOID lpOutBuffer, DWORD dwOutBufferSize,

LPDWORD lpBytesReturned);

**Parameters**

*lUserID*

[in]the return value of NET\_SDK\_Login() *lpOutBuffer*

[out]buf of the alarm event output *dwOutBufferSize*

[in]the space of lpOutBuffer applied for *lpBytesReturned*

[out]the size of the valid data of the returned lpOutBuffer

**typedef struct \_alarm\_status**

{

unsigned int iSize; //The length of the structure

int chanl; //Alarm channel.The alarm unrelated to the channel is -1.

unsigned int alarmType; //Alarm event NET\_SDK\_N9000\_ALARM\_TYPE }DD\_ALARM\_STATUS\_INFO;

enum NET\_SDK\_N9000\_ALARM\_TYPE {

NET\_SDK\_N9000\_ALARM\_TYPE\_RANGE\_BEGIN, NET\_SDK\_N9000\_ALARM\_TYPE\_MOTION=0x01,/////Motion

detection alarm input NET\_SDK\_N9000\_ALARM\_TYPE\_SENSOR,/////Sensor alarm

input

NET\_SDK\_N9000\_ALARM\_TYPE\_VLOSS,////Video loss alarm input

NET\_SDK\_N9000\_ALARM\_TYPE\_FRONT\_OFFLINE, //////Camera offline alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_OSC, removal detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD, video signal detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_SCENE,

////Object

////Abnormal

////Abnormal

video signal detection alarm for scene changed,only for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_CLARITY, ////Abnormal video signal detection alarm for video fuzzy,only for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_COLOR, ////Abnormal video signal detection alarm for color problem,only for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_TRIPWIRE, crossing detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_PERIMETER, Intrusion detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_VFD,

////Line

////Region

////Face

detection(only for ipc)

NET\_SDK\_N9000\_ALARM\_TYPE\_CDD, density

NET\_SDK\_N9000\_ALARM\_TYPE\_IPD, intrusion

NET\_SDK\_N9000\_ALARM\_TYPE\_CPC, counting

NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH,

////Crowdy

////people

////people

////face comparation alarm(for nvr) NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH\_FOR\_IPC,

////face comparation alarm(for ipc) NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_FOR\_IPC,

////Line crossing and region intrusion detection NET\_SDK\_N9000\_ALARM\_TYPE\_TRAJECT,

////target tracking trajectory NET\_SDK\_N9000\_ALARM\_TYPE\_VEHICE,

////license plate for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_AOIENTRY,

////enter region for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_AOILEAVE,

////leave region for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_PASSLINE,

////passline counting for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_SPEED\_OVER=0x21,//overspeed alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_CROSS\_BOADER,//line crossing

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_TEMPERATURE\_OVER,//temperat ure alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_X,//GSENSOR alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_Y,

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_Z,

NET\_SDK\_N9000\_ALARM\_TYPE\_EXCEPTION = 0x41,

NET\_SDK\_N9000\_ALARM\_TYPE\_IP\_CONFLICT, address conflict

NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_IO\_ERROR,

/////IP

/////Disk IO

error

NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_FULL, /////Disk full

NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_SUBHEALTH, //Raid subhealth

NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_UNAVAILABLE, //Raid unavailabe

NET\_SDK\_N9000\_ALARM\_TYPE\_ILLEIGAL\_ACCESS, /////Illegal access

NET\_SDK\_N9000\_ALARM\_TYPE\_NET\_DISCONNECT, /////Network disconnection

NET\_SDK\_N9000\_ALARM\_TYPE\_NO\_DISK, ////No disk

NET\_SDK\_N9000\_ALARM\_TYPE\_SIGNAL\_SHELTER, //Signal obstruction

NET\_SDK\_N9000\_ALARM\_TYPE\_HDD\_PULL\_OUT, //HDD pulled out

NET\_SDK\_N9000\_ALARM\_TYPE\_ALARM\_OUT = 0x51, /////Alarm output tpye.

NET\_SDK\_N9000\_ALARM\_TYPE\_RANGE\_END = 0xFF,////It is unable to exceed this value, or there will be a risk of error.

};

**Return Values**

TRUE means success, FALSE means failed. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

**Client SDK Instructions**

**NET\_SDK\_GetAlarmStatusEx**

Get the alarm information(including the raid alarm) of the device.

BOOL NET\_SDK\_GetAlarmStatusEx( LONG lUserID,

LPVOID lpOutBuffer, DWORD dwOutBufferSize, LPDWORD lpBytesReturned,

int \*exStructNum );

**Parameters**

*lUserID*

[in]the return value of NET\_SDK\_Login() *lpOutBuffer*

[out]buf of the alarm event output *dwOutBufferSize*

[in]the space of lpOutBuffer applied for *lpBytesReturned*

[out]the size of the valid data of the returned lpOutBuffer

*exStructNum*

[out]the number of the raid alarms

**typedef struct \_alarm\_status\_ex**

{

unsigned int iSize; //The length of the structure

int chanl; //Alarm channel.The alarm unrelated to the channel is -1.

unsigned int alarmType; //Alarm event NET\_SDK\_N9000\_ALARM\_TYPE, NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_SUBHEALTH,NET\_ SDK\_N9000\_ALARM\_TYPE\_RAID\_UNAVAILABLE

char alarmNode[32]; //The length of the structure

char recv[32]; //reserved. }DD\_ALARM\_STATUS\_INFO\_Ex;

enum NET\_SDK\_N9000\_ALARM\_TYPE {

NET\_SDK\_N9000\_ALARM\_TYPE\_RANGE\_BEGIN, NET\_SDK\_N9000\_ALARM\_TYPE\_MOTION=0x01,/////Motion

detection alarm input NET\_SDK\_N9000\_ALARM\_TYPE\_SENSOR,/////Sensor alarm

input

NET\_SDK\_N9000\_ALARM\_TYPE\_VLOSS,////Video loss alarm input

NET\_SDK\_N9000\_ALARM\_TYPE\_FRONT\_OFFLINE, //////Camera offline alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_OSC, removal detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD, video signal detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_SCENE,

////Object

////Abnormal

////Abnormal

video signal detection alarm for scene changed,only for

ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_CLARITY, ////Abnormal video signal detection alarm for video fuzzy,only for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_COLOR, ////Abnormal video signal detection alarm for color problem,only for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_TRIPWIRE, ////Line crossing detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_PERIMETER, ////Region Intrusion detection alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_VFD, detection(only for ipc)

NET\_SDK\_N9000\_ALARM\_TYPE\_CDD, density

NET\_SDK\_N9000\_ALARM\_TYPE\_IPD, intrusion

NET\_SDK\_N9000\_ALARM\_TYPE\_CPC, counting

NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH,

////Face

////Crowdy

////people

////people

////face comparation alarm(for nvr) NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH\_FOR\_IPC,

////face comparation alarm(for ipc) NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_FOR\_IPC,

////Line crossing and region intrusion detection NET\_SDK\_N9000\_ALARM\_TYPE\_TRAJECT,

////target tracking trajectory NET\_SDK\_N9000\_ALARM\_TYPE\_VEHICE,

////license plate for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_AOIENTRY,

////enter region for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_AOILEAVE,

////leave region for ipc NET\_SDK\_N9000\_ALARM\_TYPE\_PASSLINE,

////passline counting for ipc

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_SPEED\_OVER=0x21,//overspeed alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_CROSS\_BOADER,//line crossing

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_TEMPERATURE\_OVER,//temperat ure alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_X,//GSENSOR alarm

NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_Y, NET\_SDK\_N9000\_ALARM\_TYPE\_GPS\_GSENSOR\_Z,

NET\_SDK\_N9000\_ALARM\_TYPE\_EXCEPTION = 0x41,

NET\_SDK\_N9000\_ALARM\_TYPE\_IP\_CONFLICT, address conflict

NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_IO\_ERROR, error

/////IP

/////Disk IO

NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_FULL, /////Disk full

NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_SUBHEALTH, //Raid subhealth

NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_UNAVAILABLE, //Raid unavailabe

NET\_SDK\_N9000\_ALARM\_TYPE\_ILLEIGAL\_ACCESS, /////Illegal access

NET\_SDK\_N9000\_ALARM\_TYPE\_NET\_DISCONNECT, /////Network disconnection

NET\_SDK\_N9000\_ALARM\_TYPE\_NO\_DISK, disk

NET\_SDK\_N9000\_ALARM\_TYPE\_SIGNAL\_SHELTER,

////No

//Signal

obstruction

NET\_SDK\_N9000\_ALARM\_TYPE\_HDD\_PULL\_OUT, //HDD pulled out

NET\_SDK\_N9000\_ALARM\_TYPE\_ALARM\_OUT /////Alarm output tpye.

NET\_SDK\_N9000\_ALARM\_TYPE\_RANGE\_END

= 0x51,

= 0xFF,////It is

unable to exceed this value, or there will be a risk of error.

};

**Return Values**

TRUE means success, FALSE means failed. To get error information, please call [NET\_SDK\_GetLastError](#_page_843_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceSupportFunction**

get the functions of the IPC（only support IPC）。

BOOL NET\_SDK\_GetDeviceSupportFunction( LONG *lUserID*, [NET\_SDK\_DEV\_SUPPORT\*](#_page_786_0) pDevSupport; );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *pDevSupport*

[out] the functions of the IPC

**Return Values**

TRUE means success, FALSE means failed。To get error information, please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceIPCInfo**

get device management information

LONG NET\_SDK\_GetDeviceIPCInfo( LONG *lUserID*,

[NET\_SDK\_IPC\_DEVICE\_INFO](#_page_712_0) *\*pDeviceIPCInfo*, LONG *lBuffSize*,

LONG *\*pIPCCount* );

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *\*pDeviceIPCInfo*

[in] added IPC structural buffer *lBuffSize*

[in] size of pDeviceIPCInfo(sizeof(NET\_SDK\_IPC\_DEVICE\_INFO)\* the number of digital channel)

*\*pIPCCount*

[in] the number of IPC has been added

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](#_page_843_0)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_GetDeviceCHStatus**

Get the channel information of the NVR, like channel type, online or offline status, etc.

BOOL NET\_SDK\_GetDeviceCHStatus( LONG *lUserID*,

NET\_SDK\_CH\_DEVICE\_STATUS\* pDeviceCHStatus, LONG *lBuffSize*,

LONG *\**pCHCount );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *\*pDeviceCHStatus*

[in] the connection status of the current channel configured

*lBuffSize* [in] size of

pDeviceCHStatus(sizeof(NET\_SDK\_CH\_DEVICE\_STATU S) \* support how many channels )

*\*pCHCount*

[in] the actual numbers of the current channels

**Return Values**

TRUE means success; FALSE means failure. To get error information,please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_Login](#_page_138_0)

Client SDK Instructions

**NET\_SDK\_SetDeviceManualAlarm**

set device manual alarm

BOOL NET\_SDK\_SetDeviceManualAlarm( LONG *lUserID*,

LONG *\*pAramChannel*, LONG *\*pValue*,

LONG *lAramChannelCount* BOOL *bAlarmOpen*

);

**Parameters**

*lUserID*

[in] return value of NET\_SDK\_Login() *\*pAramChannel*

[in] List of alarm output channels,an array witch is needed to asign values,its size is **lAramChannelCount**

*\*pValue*

[in] Alarm channel status(1 means enable alarm channel,0 means disable,if all 0 means all alarm channel is disable)

*lAramChannelCount*

[in] Number of alarm channels(return value of **NET\_SDK\_DEVICEINFO::sensorOutputNum** )

*BOOL bAlarmOpen*

[in] TRUE means open alarm,FALSE means close alarm

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call[NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_GetIVMRuleConfig**

Get IVM configuration information of the device(only support IPC).

BOOL NET\_SDK\_GetIVMRuleConfig( LONG *lUserID*,

DWORD *dwCommand*, LONG *lChannel*, LPVOID *lpOutBuffer*,

DWORD *dwOutBufferSize*, LPDWORD *lpBytesReturned*,

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] the configuration command of the device. Refer to configuration command. *lChannel*

[in] the channel number starts from 0 *lpOutBuffer*

[out] a pointer to the buffer of receiving data. *dwOutBufferSize*

[in] the buffer length receiving data (in bytes) can not be zero. *lpBytesReturned*

[out] a pointer to the data length actually received can not be NULL

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

The structures and commands are as follows:

**dwCommand**

**macro definition**

**dwCommand dwCommand**

**structu value** **Meanings**

IVM\_RULE\_VFD\_CONFIG 0x0

IVM\_RULE\_VFD\_TRIGGER\_CONFIG 0x1

faceirecognitionNET\_SDK\_VFD\_CON

alarm trigger

configuration of NET\_SDK\_VFD\_TRIG face recognition

IVM\_RULE\_VFD\_SCHEDULE\_CONFIG0x2 face recognitionNET\_DVR\_SCHEDUL

IVM\_RULE\_AVD\_CONFIG 0x3 abnormal video NET\_SDK\_AVD\_CON signal detection

configuration

**See Also**

[NET\_SDK\_SetDVRConfig](clbr://internal.invalid/book/NET_SDK_SetDVRConfig.htm) [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SetIVMRuleConfig**

Set the configuration information of the device.(only support IPC)

BOOL NET\_SDK\_SetIVMRuleConfig( LONG *lUserID*,

DWORD *dwCommand*, LONG *lChannel*, LPVOID *lpInBuffer*, DWORD *dwInBufferSize* );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] the configuration command of the device. Refer to configuration command. *lChannel*

[in] the channel number starts from 0 *lpInBuffer*

[in] a pointer to the buffer of input data *dwOutBufferSize*

[in] the buffer length of the input data (in bytes)

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

The structures and commands are as follows:

**dwCommand**

**macro definition**

**dwCommand dwCommand**

**structu value** **Meanings**

IVM\_RULE\_VFD\_CONFIG 0x0

IVM\_RULE\_VFD\_TRIGGER\_CONFIG 0x1

faceirecognitionNET\_SDK\_VFD\_CON

alarm trigger

configuration of NET\_SDK\_VFD\_TRIG face recognition

IVM\_RULE\_VFD\_SCHEDULE\_CONFIG0x2 face recognitionNET\_DVR\_SCHEDUL

abnormal video IVM\_RULE\_AVD\_CONFIG 0x3 signal detection NET\_SDK\_AVD\_CON

configuration

**See Also**

NET\_SDK\_GetIVMRuleConfig [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SmartSubscrib**

Subscribe the smart alarm events.(only support IPC)

BOOL NET\_SDK\_SmartSubscrib( LONG *lUserID*,

DWORD *dwCommand*, LONG *lChannel*,

NET\_DVR\_SUBSCRIBE\_REPLY \*pOutBuffer );

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] the configuration command of the device. Refer to configuration command.

*lChannel*

[in] the channel number starts from 0 *pOutBuffer*

[out] a pointer to the buffer of input data

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

The dwCommand is as follows:

**enum definition** **value meanings**

NET\_DVR\_SMART\_AVD 0x0

NET\_DVR\_SMART\_VFD 0x1

NET\_DVR\_SMART\_VFD\_MATCH 0x2

NET\_DVR\_SMART\_PEA 0x3

NET\_DVR\_SMART\_OSC 0x4

NET\_DVR\_SMART\_CPC(obsolete) 0x5

NET\_DVR\_SMART\_CDD 0x6

NET\_DVR\_SMART\_IPD 0x7

NET\_DVR\_SMART\_VIHICLE 0x8

NET\_IPC\_SMART\_AOIENTRY 0x9

NET\_IPC\_SMART\_AOILEAVE 0xA

Abnormal video signal diagnosis

Face detection

Face comparison

Region intrusion

Object removal

People counting

Crowd density detection

People intrusion

Vehicle detection

Enter region

Leave region

Face match NET\_DVR\_SMART\_VFD\_MATCH\_FAILED0xB failed, for

stranger

NET\_IPC\_SMART\_PASSLINE 0xC pass line

**See Also**

NET\_SDK\_UnSmartSubscrib [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_UnSmartSubscrib**

Cancel the subscription of smart alarm events(only support IPC).

BOOL NET\_SDK\_UnSmartSubscrib( LONG *lUserID*,

DWORD *dwCommand*, LONG *lChannel*,

char *\*pInServerAddress*, int *\*dwResult*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] the configuration command of the device. Refer to configuration command.

*lChannel*

[in]the channel number starts from 0 *pInServerAddress*

[in] a pointer to the buffer of input data *dwResult*

[out] the length of output data

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**Remarks**

The structures and commands are as follows:

**enum definition** **value meanings**

NET\_DVR\_SMART\_AVD 0x0

NET\_DVR\_SMART\_VFD 0x1

NET\_DVR\_SMART\_VFD\_MATCH 0x2

NET\_DVR\_SMART\_PEA 0x3

NET\_DVR\_SMART\_OSC 0x4

NET\_DVR\_SMART\_CPC(obsolete) 0x5

NET\_DVR\_SMART\_CDD 0x6

NET\_DVR\_SMART\_IPD 0x7

NET\_DVR\_SMART\_VIHICLE 0x8

NET\_IPC\_SMART\_AOIENTRY 0x9

NET\_IPC\_SMART\_AOILEAVE 0xA

Abnormal video signal diagnosis

Face detection

Face comparison

Region intrusion

Object removal

People counting

Crowd density detection

People intrusion

Vehicle detection

Enter region

Leave region

Face match NET\_DVR\_SMART\_VFD\_MATCH\_FAILED0xB failed, for

stranger

NET\_IPC\_SMART\_PASSLINE 0xC pass line

**See Also**

NET\_SDK\_SmartSubscrib [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_SetSubscribCallBack**

Set the report and callback of the smart alarm events.

BOOL NET\_SDK\_SetSubscribCallBack( SUBSCRIBE\_CALLBACK *fSubscribCallBack*, void *\*pUser*,

);

**Parameters**

*fSubscribCallBack*

[in] callback function *pUser*

[in] client data

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](file:///C:/Users/Administrator/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_43164.7513164005/NET_SDK_GetlastError.htm)

Client SDK Instructions

**SUBSCRIBE\_CALLBACK**

When the subscribed smart alarm event happens, the uploading analytic data is called back

void \*SUBSCRIBE\_CALLBACK( LONG *lUserID*, DWORD *dwCommand*, char *\*pBuf*, DWORD *dwBufLen*, void *\*pUser*

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] the configuration command of the device. Refer to configuration command. *pBuf*

[in] the return data （different data types have different structures） *dwBufLen*

[out] the reture data length （different data types have different data length） *pUser*

[in] user data

**Remarks**

The dwCommand is as follows:

**dwCommand enum** **value** **mean**

NET\_SDK\_SMART\_EVENT\_TYPE\_AVD

NET\_SDK\_SMART\_EVENT\_TYPE\_VFD

NET\_SDK\_SMART\_EVENT\_TYPE\_FACE\_MATCH

6 Abnoral video sign

12 Face detection

16 Face comparison

NET\_SDK\_SMART\_EVENT\_TYPE\_FACE\_MATCH\_FOR\_IPC17 Face comparison

NET\_SDK\_SMART\_EVENT\_TYPE\_PEA\_FOR\_IPC

NET\_SDK\_SMART\_EVENT\_TYPE\_VEHICE

NET\_SDK\_SMART\_EVENT\_TYPE\_PASSLINE

18 Line crossing and

20 Vehicle number d

23 pass line

**See Also**

NET\_SDK\_UnSmartSubscrib [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FaceMatchOperate**

(only support N9000, not support IPC, IPC refer to NET\_SDK\_TransparentConfig)The relevant operation of face comparison: whether to support face comparison, face picture database management, face match alarm, getting the data of target.

BOOL NET\_SDK\_FaceMatchOperate( LONG *lUserID*,

DWORD *dwCommand*, LPVOID *lpInBuffer*, DWORD *dwInBufferSize*, LPVOID *lpOutBuffer*, DWORD *dwOutBufferSize*, LPDWORD *lpBytesReturned*,

);

**Parameters**

*lUserID*

[in] the return value of NET\_SDK\_Login() *dwCommand*

[in] Command types refer to configuration command *lpInBuffer*

[in] a buffer pointer to send data *dwInBufferSize*

[in] the buffer size of sending data (in bytes) *lpOutBuffer*

[out] a buffer pointer to receive data *dwOutBufferSize*

[in] the buffer size of receiving data (in bytes) *lpBytesReturned*

[out] the data length pointer that actually receives can not larger than dwOutBufferSize

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/NET_SDK_GetlastError.htm)

**Remarks**

Different functions have different structure and commands as shown below.

**dwCommand** **dwCommanddwCommand**

**Macro Definition** **Value**

NET\_SDK\_GET\_FACE\_MATCH\_SUPPORT 0x01

**Definition**

Whether to support face comparison or not

NET\_SDK\_GET\_FACE\_INFO\_GROUP\_LIST 0x02 Get group list NULL

NET\_SDK\_ADD\_FACE\_INFO\_GROUP 0x03 NET\_SDK\_SET\_FACE\_INFO\_GROUP 0x04 NET\_SDK\_DEL\_FACE\_INFO\_GROUP 0x05

NET\_SDK\_GET\_FACE\_INFO\_LIST 0x06

NET\_SDK\_ADD\_FACE\_INFO 0x07

NET\_SDK\_SET\_FACE\_INFO 0x08

NET\_SDK\_DEL\_FACE\_INFO 0x09

NET\_SDK\_GET\_FACE\_MATCH\_ALARM 0x0A

NET\_SDK\_SET\_FACE\_MATCH\_ALARM 0x0B

NET\_SDK\_GET\_FACE\_INFO\_IMG 0x0C

NET\_SDK\_SEARCH\_IMAGE\_BY\_IMG 0x0D

Add group Edit group Delete group

Get target face list

Add target face

Edit face information

Delete face picture

Get face match alarm linkage

Set face match alarm linkage

Get target face data

Search image by image

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_GROUP_ADD.htm) [NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_GROUP_ITEM.htm) [NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_GROUP_DEL.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_LIST_GET.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_ADD.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_EDIT.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_DEL.htm)

NULL

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_MATCH_ALARM.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_INFO_IMG_GET.htm)

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_SEARCH_IMAGE_BY_IMAGE.htm)

NET\_SDK\_SEARCH\_CH\_SNAP\_FACE\_IMG\_LIST0x0E

Search faces [NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_CH_SNAP_FACE_IMG_LIST_SEARCH.htm) of the camera [H](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_CH_SNAP_FACE_IMG_LIST_SEARCH.htm)

NET\_SDK\_SEARCH\_CH\_SNAP\_FACE\_IMG 0x0F

Search the face information of the camera

[NET\_SDK\_](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43440.705677338/interface/struct/NET_SDK_FACE_IMG_INFO_CH.htm)

Client SDK Instructions

**NET\_SDK\_SetFishEyeAdjust**

(windows only)Set the fisheye correction mode，

This interface is called to enter and exit fisheye correct ion mode both, only in single window mode.

BOOL NET\_SDK\_SetFishEyeAdjust( POINTERHANDLE lPlayHandle, FISHEYE\_MODE fishEyeMode

);

**Parameters**

*lPlayHandle*

[in] the handel of play video *fishEyeMode*

[in]

fisheye mode:Installation mode + correction mode, The specifi c defined values are as follows:

typedef enum {

FISHEYE\_ORIGNAL = 0, //Original mode ,That is, the fisheye map in the top / wall / botto m mount mode, equivalent to quitting the fisheye

correction mod

FISHEYE\_ROOF = 0x0100, //Top (suction top) FISHEYE\_ROOF\_360, //Top-

mounted 360 rectangular expansion panorama + independent sub-screen;Sub-

frames and rectangularexpansion panorama suppor t doubling and moving operations，

Rectangular expansion panorama also supports le ft and right start point movement

FISHEYE\_ROOF\_2x180, //Two associated 180 re ctangular expansion screens of top-mounted，

At any moment, the two sub-

Windows constitute 360 panoramic views, also kn own as the "double panorama",Both rectangular e xpansion pictures support the left and right move ment start point operation, and linkage with each other;

FISHEYE\_ROOF\_FISH\_3PTZ, //Top-mounted original image + 3 independent sub-images，Both sub-

frames and frames in the original image support d oubling and moving，

The original image also supports rotation change start point operations;

FISHEYE\_ROOF\_FISH\_4PTZ, //Top-mounted original image + 4 independent subimages， Both sub-

frames and frames in the original image support d oubling and moving，

The original image also supports rotation change start point operations;

FISHEYE\_ROOF\_360\_6PTZ, //Top-

mounted 360 rectangular expansion panorama +6 independent sub-screens，Sub-

frames and rectangular expansion panorama suppor t doubling and moving operations，

Rectangular expansion panorama also supports le

ft and right start point movement FISHEYE\_ROOF\_FISH\_8PTZ, //Top-

mounted original image + 8 independent sub-images，Both sub-

frames and frames in the original image support d oubling and moving，

The original image also supports rotation change start point operations;

FISHEYE\_WALL = 0x0200, //Wall-mounted FISHEYE\_WALL\_180, //The 180 wall-

mounted panorama, from left to right 180 rect angular expansion panorama, 180 rectangular expansion panorama support up and down mo vement operation, change the vertical viewing angle;

FISHEYE\_WALL\_180\_3PTZ, //180 Rectangle panoramic panorama of wall-

mounted+ 3independent sub-frames，sub-

frames and rectangular panoramic panorama s upport doubling and moving operations， Rectangular expansion panorama supports up and down movement,to change the vertical per spective

FISHEYE\_WALL\_180\_4PTZ, //180 Rectangle panoramic panorama of wall-

mounted+4 independent sub-frames，sub-

frames and rectangular panoramic panorama s upport doubling and moving operations， Rectangular expansion panorama supports up and down movement,to change the vertical per spective

FISHEYE\_WALL\_180\_8PTZ, //180 Rectangl e panoramic panorama of wall-

mounted+ 8 independent sub-frames，sub-

frames and rectangular panoramic panorama s upport doubling and moving operations， Rectangular expansion panorama supports up and down movement,to change the vertical per spective

FISHEYE\_DESKTOP = 0x0300, //Bottom-mounted(desktop)

FISHEYE\_DESKTOP\_360, // 360 rectangula r expansion panorama of bottom-mounted+independent sub-frames;sub-

frames and rectangular expansion panorama sup port doubling and moving operations， Rectangular expansion panorama also support s left and right start point movement

FISHEYE\_DESKTOP\_2x180, //Two associate d 180 rectangular expansion screens of bottom-mounted，At any moment, the two sub-

Windows constitute 360 panoramic views, also known as the "double panorama",Both rectan

gular expansion pictures support the left and ri ght movement start point operation, and linka ge with each other;

FISHEYE\_DESKTOP\_FISH\_3PTZ, //Bottom-mounted original image + 3 independent sub-images，Both sub-

frames and frames in the original image suppo rt doubling and moving，

The original image also supports rotation chan

ge start point operations; FISHEYE\_DESKTOP\_FISH\_4PTZ, //Bottom-

mounted original image + 4 independent sub-images，Both sub-

frames and frames in the original image suppo rt doubling and moving，

The original image also supports rotation chan ge start point operations;

FISHEYE\_DESKTOP\_360\_6PTZ, //Bottom-mounted 360 rectangular expansion panorama

+6 independent sub-screens，Sub-

frames and rectangular expansion panorama sup port doubling and moving operations， Rectangular expansion panorama also support s left and right start point movement

FISHEYE\_DESKTOP\_FISH\_8PTZ, //Bottom-mounted original image + 8 independent sub-images，Both sub-

frames and frames in the original image suppo rt doubling and moving，

The original image also supports rotation chan ge start point operations;

}FISHEYE\_MODE;

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FishEyeAdjustFocus**

(windows only)set focus，

to identify which segmentation region of the current ac tion acts on fisheye correction

BOOL NET\_SDK\_FishEyeAdjustFocus( POINTERHANDLE lPlayHandle,

int nX, int nY

);

**Parameters**

*lPlayHandle*

[in] the handel of play video *nX*

[in] the X coordinate value of current focus,relative to the coordinate system of current play window

nY

[in] the Y coordinate value of current focus,relative to the coordinate system of current play window

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FishEyeAdjustFocusEx**

(windows only)set focus，

to identify which segmentation region of the current ac tion acts on fisheye correction

BOOL NET\_SDK\_FishEyeAdjustFocusEx( POINTERHANDLE lPlayHandle,

int nX, int nY,

int &nIndex );

**Parameters**

*lPlayHandle*

[in] the handel of play video *nX*

[in] the X coordinate value of current focus,relative to the coordinate system of current play window

nY

[in] the Y coordinate value of current focus,relative to the coordinate system of current play window

nIndex

[in] fccus index

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FishEyeAdjustMove**

(windows only)E-cloud platform movement,the segmentation belong to the e-cloud platform can be moved only when it's under the fisheye correction mode

BOOL NET\_SDK\_FishEyeAdjustMove( POINTERHANDLE lPlayHandle,

int nMoveX, int nMoveY

);

**Parameters**

*lPlayHandle*

[in] the handel of play video *nMoveX*

[in] the left mouse button drags horizontally against the X axis of the starting point,positive to the right and negative to the left,with the starting point as the origin

nMoveY

[in] the left mouse button drags vertically against the Y axis of the starting point,positive up and negative down,with the starting point as the origin

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FishEyeAdjustGetArea**

(windows only)obtain the correction area location of the current focus

BOOL NET\_SDK\_FishEyeAdjustGetArea( POINTERHANDLE lPlayHandle,

RECT &AreaRect );

**Parameters**

*lPlayHandle*

[in] the handel of play video *AreaRect*

[in] the correction area location of the current focus,relative to the current play window coordinate system

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**NET\_SDK\_FishEyeAdjustZoom**

(windows only)E-cloud platform amplification,the division belongs to the e-cloud platform can be amplified only when it's into the fisheye correction mode

BOOL NET\_SDK\_FishEyeAdjustZoom( POINTERHANDLE lPlayHandle,

const RECT &ZoomRect );

**Parameters**

*lPlayHandle*

[in] the handel of play video *ZoomRect*

[in] Specifies the area location information to zoom in,relative to the current play window coordinate system

**Return Values**

TRUE means success; FALSE means failure. To get error information, please call [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

Client SDK Instructions

**DD\_ACCOUNT\_CONFIG**

struct of account configuration

struct \_dd\_account\_config{ unsigned long iSize; unsigned long enable; unsigned long bindMAC; unsigned long group; char MAC [8];

char name [DD\_MAX\_USER\_NAME\_BUF\_LEN]; char password [DD\_MAX\_PASSWORD\_BUF\_LEN]; unsigned char logSearch;

unsigned char systemSetup; unsigned char fileManagement; unsigned char diskManagement; unsigned char remoteLogin; unsigned char twoWayAudio; unsigned char systemMaintain;

unsigned char OnlineUserManagement; unsigned char shutdown;

unsigned char alarmOutCtrl; unsigned char netAlarm; unsigned char netSerialCtrl; unsigned char authLive; unsigned char authRecord; unsigned char authPlayback; unsigned char authBackup; unsigned char authPTZ; unsigned char netAuthView; unsigned char netauthRecord; unsigned char netauthPlayback; unsigned char netauthBackup; unsigned char netauthPTZ; unsigned char recv[2];

unsigned char authLiveCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char authRecordCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char authPlaybackCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char authBackupCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char authPTZCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char netAuthViewCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char netAuthRecordCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char netAuthPlaybackCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char netAuthBackupCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char netAuthPTZCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; }DD\_ACCOUNT\_CONFIG;

**Members**

*iSize*

size of the struct *enable*

whether to use the account *bindMAC*

whether to bind MAC *group*

the group belonged to,refer to DD\_USER\_GROUP:

**Group Name** **Value** **Description**

DD\_USER\_GROUP\_NONE 0x00

DD\_USER\_GROUP\_ADMIN 0x01 administrator,have all rights

DD\_USER\_GROUP\_ADVANCE0x02

advanced user,default rights:basic,record,config,playback,backup,data management,disk management,PTZ control,remote login and all channels rights

normal user,default DD\_USER\_GROUP\_NORMAL 0x03 rights:basic,record,playback,backup,PTZ

control,remote login and all channels rights

*MAC [8]* binded MAC

*name [DD\_MAX\_USER\_NAME\_BUF\_LEN]* user name

*password [DD\_MAX\_PASSWORD\_BUF\_LEN]* password

*logSearch*

limit of log search *systemSetup*

system configuration *fileManagement*

file management *diskManagement*

disc management *remoteLogin*

remote login *twoWayAudio*

audio talkback *systemMaintain*

system maintain *OnlineUserManagement*

online user management *shutdown*

shutdown or reboot *alarmOutCtrl*

alarm output control *netAlarm*

network alarm *netSerialCtrl*

network serial port control *authLive*

live preview *authRecord*

local record *authPlayback*

local search playback *authBackup*

local backup *authPTZ*

local PTZ *netAuthView*

local control

*netauthRecord* remote record

*netauthPlayback* remote live playback

*netauthBackup* remote backup

*netauthPTZ* remote PTZ

*recv[2]* reserved bytes

*authLiveCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* live preview channel

*authRecordCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* local record manually

*authPlaybackCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* local search and playback

*authBackupCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* local backup

*authPTZCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* local PTZ control

*netAuthViewCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* remote live preview

*netAuthRecordCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* remote record manually

*netAuthPlaybackCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* remote playback

*netAuthBackupCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* remote backup

*netAuthPTZCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* remote PTZ control

Client SDK Instructions

**DD\_AREA**

area struct

struct \_dd\_area\_{ unsigned short x; unsigned short y; unsigned short cx; unsigned short cy; }DD\_AREA;

**Members**

*x*

abscissa,range 0-99 *y*

ordinate,range 0-99 *cx*

width,range 1-100 *cy*

height,range 1-100

**Notice:** x+cx <= 100, y+cy <=100

Client SDK Instructions

**DD\_AUTO\_REPORT**

Passively receive the struct of DVR register.

struct \_dd\_auto\_report\_{ unsigned long bUse; char host[256]; unsigned long dwPort; unsigned long ID; }DD\_AUTO\_REPORT;

**Members**

*bUse*

Whether to enable auto report register function *host[256]*

server address of register platform *dwPort*

server port of register platform *ID*

assigned register ID

Client SDK Instructions

**DD\_BASIC\_CONFIG**

struct of basic configuration information

struct \_dd\_basic\_config\_{ unsigned long iSize; unsigned long videoFormat; unsigned long videoOut;

unsigned long videoOutResolution; unsigned long VGARefresh; unsigned long screensaver; unsigned long deviceLanguage; unsigned long passwordCheck; unsigned long RecycleRecord; unsigned long videoFormatMask; unsigned long videoOutMask;

unsigned long videoOutResolutionMask; unsigned long languageMask; }DD\_BASIC\_CONFIG;

**Members**

*iSize*

size of the struct *videoFormat*

video format, refer to DD\_VIDEO\_FORMAT:

**Type** **Value**

DD\_VIDEO\_FORMAT\_NTSC 0x01 DD\_VIDEO\_FORMAT\_PAL 0x02

*videoOut*

video output device(reserved) *videoOutResolution*

video output resolution,refer to DD\_VGA\_RESOLUTION:

**Type** **Value**

DD\_VGA\_640X480 0x0001 DD\_VGA\_720X480 0x0002 DD\_VGA\_720X576 0x0004 DD\_VGA\_800X600 0x0008 DD\_VGA\_1024X768 0x0010 DD\_VGA\_1280X960 0x0020 DD\_VGA\_1280X1024 0x0040 DD\_VGA\_1920X1080 0x0080

*VGARefresh*

VGA refresh rate(reserved) *screensaver*

screensaver time(0 means close) *deviceLanguage*

device language *passwordCheck*

whether to open password check *RecycleRecord*

whether permit overlap record *videoFormatMask*

supportive video format mask(read only) *videoOutMask*

supportive video output device mask(read only) *videoOutResolutionMask*

supportive video output device resolution mask(read only)

*languageMask*

language mask group supported by device(read only),refer to the list below:

类型 对应值

LANGUAGE\_ENGLISH LANGUAGE\_CHINESE\_S LANGUAGE\_CHINESE\_B LANGUAGE\_PORTUGUESE LANGUAGE\_GREECE LANGUAGE\_SPAISH LANGUAGE\_RUSSIAN LANGUAGE\_NORWAY LANGUAGE\_TURKEY LANGUAGE\_ITALY LANGUAGE\_CZECH LANGUAGE\_GERMAN LANGUAGE\_HEBREW LANGUAGE\_JAPANESE LANGUAGE\_FRENCH LANGUAGE\_POLISH LANGUAGE\_BULGARIAN LANGUAGE\_INDONESIA LANGUAGE\_RUSSIAN\_D LANGUAGE\_THAI LANGUAGE\_HUNGARY

LANGUAGE\_LITHUANIA

0x0000001 0x0000002 0x0000004 0x0000008 0x0000010 0x0000020 0x0000040 0x0000080 0x0000100 0x0000200 0x0000400 0x0000800 0x0001000 0x0002000 0x0004000 0x0008000 0x0010000 0x0020000 0x0040000 0x0080000 0x0100000

0x0200000

Client SDK Instructions

**DD\_BUZZER\_CONFIG**

struct of buzzer configuration

struct \_dd\_buzzer\_config\_{ unsigned char

unsigned char unsigned short }DD\_BUZZER\_CONFIG;

enable; recv; holdTime;

**Members**

*enable*

buzzer enable switch *recv*

reserved bytes *holdTime*

delay time

Client SDK Instructions

**DD\_CHANNEL\_CONFIG**

struct of channel configuration information

struct \_dd\_channel\_config\_{ unsigned long iSize; unsigned long hide;

char name [DD\_MAX\_CAMERA\_NAME\_BUF\_LEN]; }DD\_CHANNEL\_CONFIG;

**Members**

*iSize*

size of the struct *hide*

whether hide channel

*name [DD\_MAX\_CAMERA\_NAME\_BUF\_LEN]* channel name

Client SDK Instructions

**DD\_CRUISE\_POINT\_INFO**

struct of setting cruise position information.

struct \_dd\_cruise\_point\_info{ unsigned long

unsigned long unsigned long

}DD\_CRUISE\_POINT\_INFO;

presetIndex; dwellSpeed; dwellTime;

**Members**

*presetIndex*

index of cruise position(1-128). *dwellSpeed*

speed of cruise(1-8). *dwellTime*

seconds of cruise .

Client SDK Instructions

**DD\_DATE**

struct of date configuration infomation of device.

struct \_dd\_date\_{ unsigned char mday; unsigned char month; unsigned short year; }DD\_DATE, \*LP\_DD\_DATE;

**Members**

*mday*

day of month, range(1-31). *month*

month, range(1-12). *year*

current solar year.

Client SDK Instructions

**DD\_DATE\_SCHEDULE**

struct of data schedule

struct \_dd\_date\_schedule\_{

unsigned long long hour [24]; }DD\_DATE\_SCHEDULE;

**Members**

*hour [24]*

data schedule formation,24 stands for 24 hours format,each position of unsigned long long stands for each minute's state

Client SDK Instructions

**DD\_DATE\_TIME\_CONFIG**

struct of data and time schedule configuration

struct \_dd\_date\_time\_config{ unsigned long iSize; unsigned char dateFormat; unsigned char timeFormat; unsigned char timeZone; unsigned char enableNTP; unsigned short ntpPort; unsigned short recv;

char ntpServerAddr[DD\_MAX\_URL\_BUF\_LEN];

}DD\_DATE\_TIME\_CONFIG;

**Members**

*iSize*

size of the struct *dateFormat*

data format, refer to DD\_DATE\_MODE:

**Type**

DD\_DATE\_MODE\_YMD DD\_DATE\_MODE\_MDY

DD\_DATE\_MODE\_DMY

**Value** **Description**

0x01 YMD format 0x02 MDY format

0x03 DMY format

*timeFormat*

time format,refer to the list below:

**Type**

TIME\_MODE\_12

TIME\_MODE\_24

**Value** **Description**

0x01 12 hours

0x02 24 hours

*timeZone*

time zone, refer to DD\_TIME\_ZOME\_NAME:

**Type** **Value**

DD\_TIME\_ZONE\_GMT\_D12 0 DD\_TIME\_ZONE\_GMT\_D11 1 DD\_TIME\_ZONE\_GMT\_D10 2 DD\_TIME\_ZONE\_GMT\_D9 3 DD\_TIME\_ZONE\_GMT\_D8 4 DD\_TIME\_ZONE\_GMT\_D7 5 DD\_TIME\_ZONE\_GMT\_D6 6 DD\_TIME\_ZONE\_GMT\_D5 7 DD\_TIME\_ZONE\_GMT\_D4\_30 8 DD\_TIME\_ZONE\_GMT\_D4 9 DD\_TIME\_ZONE\_GMT\_D3\_30 10 DD\_TIME\_ZONE\_GMT\_D3 11 DD\_TIME\_ZONE\_GMT\_D2 12 DD\_TIME\_ZONE\_GMT\_D1 13 DD\_TIME\_ZONE\_GMT 14 DD\_TIME\_ZONE\_GMT\_A1 15 DD\_TIME\_ZONE\_GMT\_A2 16 DD\_TIME\_ZONE\_GMT\_A3 17 DD\_TIME\_ZONE\_GMT\_A3\_30 18 DD\_TIME\_ZONE\_GMT\_A4 19 DD\_TIME\_ZONE\_GMT\_A4\_30 20 DD\_TIME\_ZONE\_GMT\_A5 21 DD\_TIME\_ZONE\_GMT\_A5\_30 22 DD\_TIME\_ZONE\_GMT\_A5\_45 23 DD\_TIME\_ZONE\_GMT\_A6 24 DD\_TIME\_ZONE\_GMT\_A6\_30 25

DD\_TIME\_ZONE\_GMT\_A7 26 DD\_TIME\_ZONE\_GMT\_A8 27 DD\_TIME\_ZONE\_GMT\_A9 28 DD\_TIME\_ZONE\_GMT\_A9\_30 29 DD\_TIME\_ZONE\_GMT\_A10 30 DD\_TIME\_ZONE\_GMT\_A11 31 DD\_TIME\_ZONE\_GMT\_A12 32 DD\_TIME\_ZONE\_GMT\_A13 33

*enableNTP*

whether open NTP synchronization service *ntpPort*

NTP port *ntpServerAddr[DD\_MAX\_URL\_BUF\_LEN]*

NTP service address

Client SDK Instructions

**DD\_DAYLIGHT\_INFO**

struct of daylight saving time information

struct \_dd\_daylight\_info\_{ unsigned char InMonth; unsigned char InMday; unsigned char OutMonth; unsigned char OutMday;

unsigned char InWeekIndex; unsigned char InWday; unsigned char OutWeekIndex; unsigned char OutWday; unsigned short InYear; unsigned short OutYear; unsigned short enable; unsigned short type; unsigned long InSecond; unsigned long OutSecond; unsigned long offSet; }DD\_DAYLIGHT\_INFO;

**Members**

*InMonth*

which month to enter DST *InMday*

which day to enter DST(data mode is valid) *OutMonth*

which month to exit DST *OutMday*

which day to exit DST(data mode is valid) *InWeekIndex*

which week to enter DST(week mode is valid) *InWday*

which weekday to enter DST(week mode is valid) *OutWeekIndex*

which week to exit DST(week mode is valid) *OutWday*

which weekday to exit DST(week mode is valid) *InYear*

which year to enter DST,reserved due to align the struct

*OutYear*

which year to exit DST,reserved due to align the struct *enable*

whether enable DST function *type*

DST setting modern: week or data mode *InSecond*

second offset in one day of DST(0-86399),it can switch to be hour and minute and second

*OutSecond*

second offset out of one day of DST(0-86399),it can switch to be hour and minute and second

*offSet*

offset second in DST(0-86399)

Client SDK Instructions

**DD\_DDNS\_CONFIG**

struct of DDNS configuration

struct \_dd\_ddns\_config\_{ unsigned long iSize; unsigned short enable;

unsigned short interval; unsigned long useDDNSServer; unsigned long userHostDomain; char userName

[DD\_MAX\_DDNS\_ACCOUNT\_BUF\_LEN]; char password

[DD\_MAX\_PASSWORD\_BUF\_LEN];

char hostDomain [DD\_MAX\_URL\_BUF\_LEN]; }DD\_DDNS\_CONFIG;

**Members**

*iSize*

size of the struct *enable*

whether enable DDNS *interval*

report upgrade interval *useDDNSServer*

type or address of DDNS server in use *userHostDomain*

whether enable host domain name

*userName [DD\_MAX\_DDNS\_ACCOUNT\_BUF\_LEN]* DDNS account

*password [DD\_MAX\_PASSWORD\_BUF\_LEN]* DDNS password

*hostDomain [DD\_MAX\_URL\_BUF\_LEN]*

host domain name(correspond to a certain protocol,specifying server is permitted)

Client SDK Instructions

**DD\_DDNS\_SERVER\_INFO**

struct of DDNS server information

struct \_dd\_ddns\_server\_info{ unsigned char DDNSID;

unsigned char supportproperty; unsigned char noused[2];

char DDNSServerName[64]; }DD\_DDNS\_SERVER\_INFO;

**Members**

*DDNSID*

DDNS ID,server name is valid only if ID value is greater than 0

*supportproperty* NCFG\_ENUM\_DDNS\_SUPPORT\_DOMAIN1 =0x01(support domain 1,maybe need to support two domains)

*noused[2]*

unenable DDNS server *DDNSServerName[64]*

address of DDNS server

Client SDK Instructions

**DD\_DEVICEINFO**

struct of device basic information.

struct \_dd\_device\_info\_{ unsigned long iSize; unsigned long deviceID;

char deviceNo[DD\_MAX\_SERIAL\_NUMBER\_LEN]; char deviceName [DD\_MAX\_NAME\_LEN];

char firmwareVersion [DD\_MAX\_VERSION\_BUF\_LEN];

char firmwareBuildDate [DD\_MAX\_VERSION\_BUF\_LEN];

char hardwareVersion [DD\_MAX\_VERSION\_BUF\_LEN];

char kernelVersion [DD\_MAX\_VERSION\_BUF\_LEN];

char mcuVersion [DD\_MAX\_VERSION\_BUF\_LEN]; unsigned char audioNum;

unsigned char localVideoInNum; unsigned char netVideoInNum; unsigned char sensorInNum; unsigned char relayOutNum; unsigned char rs232Num; unsigned char rs485Num; unsigned char networkPortNum; unsigned char diskCtrlNum; unsigned char DiskNum; unsigned char vgaNum;

unsigned char usbNum; }DD\_DEVICE\_INFO;

**Members**

*iSize*

size of the struct. *deviceID*

device ID(0~255). *deviceNo[DD\_MAX\_SERIAL\_NUMBER\_LEN]*

serial number of device,letter is usable. *deviceName [DD\_MAX\_NAME\_LEN]*

device name(attention to double byte character). *firmwareVersion [DD\_MAX\_VERSION\_BUF\_LEN]*

software version number.

*firmwareBuildDate [DD\_MAX\_VERSION\_BUF\_LEN]* software building date.

*hardwareVersion [DD\_MAX\_VERSION\_BUF\_LEN]* hardware version.

*kernelVersion [DD\_MAX\_VERSION\_BUF\_LEN]* system core version.

*mcuVersion [DD\_MAX\_VERSION\_BUF\_LEN]* MCU version.

*audioNum*

audio number. *localVideoInNum*

channel number of local video input. *netVideoInNum*

channel number of network video input. *sensorInNum*

number of sensor for input. *relayOutNum*

number of relay for output. *rs232Num*

channel number of 232 remote sensing. *rs485Num*

channel number of 485 remote sensing. *networkPortNum*

number of network port. *diskCtrlNum*

number of harddisk for control. *DiskNum*

number of harddisk. *vgaNum*

number of displayer. *usbNum*

number of USB soket.

Client SDK Instructions

**DD\_ENCODE\_CONFIG**

struct of encoding configuration

struct \_dd\_encode\_config\_{ unsigned long iSize; unsigned short resolution; unsigned short rate; unsigned short encodeType; unsigned short quality; unsigned short minBitrate; unsigned short maxBitrate; }DD\_ENCODE\_CONFIG;

**Members**

*iSize*

size of the struct *resolution*

resolution, refer to DD\_VIDEO\_SIZE:

**Type** **Value**

DD\_VIDEO\_SIZE\_QCIF 0x0001 DD\_VIDEO\_SIZE\_CIF 0x0002 DD\_VIDEO\_SIZE\_HD1 0x0004 DD\_VIDEO\_SIZE\_D1 0x0008 DD\_VIDEO\_SIZE\_QVGA 0x0010 DD\_VIDEO\_SIZE\_VGA 0x0020 DD\_VIDEO\_SIZE\_XVGA 0x0040 DD\_VIDEO\_SIZE\_QQVGA 0x0080 DD\_VIDEO\_SIZE\_480P 0x0100 DD\_VIDEO\_SIZE\_720P 0x0200

DD\_VIDEO\_SIZE\_1080P 0x0400

**Video Format**

QCIF CIF HD1 D1 QVGA VGA XVGA QQVGA 480P 720P

1080P

*rate*

frame rate *encodeType*

encoding type,refer to the following list: *rate*

frame rate *encodeType*

encoding type,refer to the following list:

**type** **ValueDescription**

DD\_VIDEO\_ENCODE\_MODE\_VBR0x01

DD\_VIDEO\_ENCODE\_MODE\_CBR0x02

mutable code stream

fixed code stream

*quality*

image quality, refer to DD\_IMAGE\_QUALITY:

**Type** **ValueDescription**

DD\_IMAGE\_QUALITY\_LOWEST 0x01

DD\_IMAGE\_QUALITY\_LOWER 0x02

DD\_IMAGE\_QUALITY\_LOW 0x03

DD\_IMAGE\_QUALITY\_MEDIUM 0x04

DD\_IMAGE\_QUALITY\_HEIGHTER 0x05

lowest image quality

lower image quality

low image quality

medium image quality

heighter image quality

heightest DD\_IMAGE\_QUALITY\_HEIGHTEST0x06 image

quality

*minBitrate*

code stream lower limit,in kbps *maxBitrate*

code stream upper limit,in kbps

Client SDK Instructions

**DD\_ENCODE\_CONFIG\_SUPPORT**

struct of encode config supported by device

struct \_dd\_encode\_config\_support\_ {

[DD\_ENCODE\_CONFIG](#_page_569_0) encodeConfig[DD\_MAX\_SUPPORT\_RESOLUTION];

unsigned long num; }DD\_ENCODE\_CONFIG\_SUPPORT;

**Members**

*encodeConfig[DD\_MAX\_SUPPORT\_RESOLUTION]* DD\_MAX\_SUPPORT\_RESOLUTION equals 7, which is the max supported resolution type num, encodeConfig includes main stream(sub stream) resolution, fps, max and min bitrate.

*num*

the real supported resolution num by device

Client SDK Instructions

**DD\_FRAME\_INFO**

struct of data frame information

struct \_dd\_frame\_info\_{

unsigned long frameType; unsigned long length; unsigned long keyFrame; unsigned long width; unsigned long height; unsigned long \*pData; unsigned short deviceIndex; unsigned short channel; unsigned long bufIndex; unsigned long frameIndex; unsigned long frameAttrib; unsigned long streamID; LONGLONG time;

LONGLONG relativeTime; DD\_TIME localTime; }DD\_FRAME\_INFO, \*LP\_DD\_FRAME\_INFO;

**Members**

*frameType*

data frame type, refer to DD\_FRAME\_TYPE:

**Type** **Value**

DD\_FRAME\_TYPE\_NONE 0x00 DD\_FRAME\_TYPE\_VIDEO 0x01 DD\_FRAME\_TYPE\_AUDIO 0x02 DD\_FRAME\_TYPE\_TALK\_AUDIO 0x03 DD\_FRAME\_TYPE\_JPEG 0x04 DD\_FRAME\_TYPE\_VIDEO\_FORMAT 0x05 DD\_FRAME\_TYPE\_AUDIO\_FORMAT 0x06 DD\_FRAME\_TYPE\_TALK\_AUDIO\_FORMAT0x07 DD\_FRAME\_TYPE\_END

*length*

data length *keyFrame*

keyframe,0:non-key-frame,1:keyframe *width*

width of data frame *height*

height of data frame *\*pData*

pointer to data *deviceIndex*

device index number *channel*

data channel *bufIndex*

buffer area index *frameIndex*

data frame index *frameAttrib*

data frame attribute, refer to DD\_FRAME\_ATTRIB:

**Type** **Value Description**

DD\_PLAY\_FRAME\_NO\_SHOW 0x01

DD\_PLAY\_FRAME\_SHOW 0x02

DD\_PLAY\_FRAME\_ALL\_END 0x04

DD\_PLAY\_FRAME\_SEC\_END 0x08

no show this frame

the frame can be showed

reading data is finished,no more data

the event section is ended

the frame includes

DD\_PLAY\_FRAME\_NO\_TIME\_STAMP 0x10 timestamp,shield time function when capture

DD\_PLAY\_FRAME\_FF

DD\_LIVE\_FRAME\_FIRST\_STREAM

the frame 0x20 applied to

fastforward

0x40 the frame is live main code

DD\_LIVE\_FRAME\_SECOND\_STREAM0x80

stream

the frame is live sub code stream

DD\_LIVE\_FRAME\_JPEG

DD\_LIVE\_FRAME\_TALK

the frame is JPEG image

the frame is 0x200talkback audio

data

*streamID*

data stream ID *time*

absolute time,calculate from 00:00:00 on Jan.1st in 1970,in microsecond,it changes when change device time

*relativeTime*

relative time,in microsecond,it won't change when change device time,because it is continuous

*localTime*

device local time, later fill in

Client SDK Instructions

**DD\_LIVE\_AUDIO\_GROUP**

struct of audio group

struct \_dd\_live\_audio\_group\_{ unsigned short holdTime; unsigned char volume; unsigned char channel; }DD\_LIVE\_AUDIO\_GROUP;

**Members**

*holdTime*

hold time(in second), 0 means invalid *volume*

volume(0-100) *channel*

channel number,start from 0

Client SDK Instructions

**DD\_LIVE\_DISPLAY**

struct of real time display

struct \_dd\_live\_display\_{ unsigned long iSize; unsigned long showTime; unsigned long showNetwork; unsigned long showHDD; unsigned long showUSB; unsigned short alarmInNum; unsigned short alarmOutNum;

unsigned long showAlarmIn; unsigned long showAlarmOut; unsigned long cameraNum;

unsigned char showCameraName [DD\_MAX\_CAMERA\_NUM]; unsigned char showRecordStatus [DD\_MAX\_CAMERA\_NUM]; }DD\_LIVE\_DISPLAY;

**Members**

*iSize*

size of the struct *showTime*

whether show system time *showNetwork*

whether show network status *showHDD*

whether show harddisc information *showUSB*

whether show movable storage infomation *alarmInNum*

alarm input number(read only) *alarmOutNum*

alarm output number(read only) *showAlarmIn*

whether show alarm input information *showAlarmOut*

whether show alarm output information *cameraNum*

valid channel number(read only) *showCameraName [DD\_MAX\_CAMERA\_NUM]*

whether show channel name *showRecordStatus [DD\_MAX\_CAMERA\_NUM]*

whether show record status

Client SDK Instructions

**DD\_LIVE\_VIDEO\_GROUP**

struct of preview video group

struct \_dd\_live\_video\_group\_{ unsigned short holdTime; unsigned short channelNum; unsigned long splitMode;

unsigned char channel [DD\_MAX\_CAMERA\_NUM]; }DD\_LIVE\_VIDEO\_GROUP;

**Members**

*holdTime*

hold time,(in second), 0 means invalid *channelNum*

valid channel number(read only) *splitMode*

split mode, refer to DD\_VIEW\_SPLIT\_MODE:

**type**

DD\_VIEW\_SPLIT\_1X1 DD\_VIEW\_SPLIT\_2X2

DD\_VIEW\_SPLIT\_1A2/ DD\_VIEW\_SPLIT\_2X3 DD\_VIEW\_SPLIT\_1A5/DD\_VIEW\_SPLIT\_3X3 DD\_VIEW\_SPLIT\_1A7/DD\_VIEW\_SPLIT\_1A12/DD\_VIEW\_SPLIT\_4X4 DD\_VIEW\_SPLIT\_2A6/DD\_VIEW\_SPLIT\_4X6 DD\_VIEW\_SPLIT\_1A9/DD\_VIEW\_SPLIT\_4A9/DD\_VIEW\_SPLIT\_1A16/DD\_VIEW\_SPL DD\_VIEW\_SPLIT\_1A11/DD\_VIEW\_SPLIT\_1A20/DD\_VIEW\_SPLIT\_4A20/DD\_VIEW\_S

*channel [DD\_MAX\_CAMERA\_NUM]*

channel number corresponding to each area,array index stands for channel number,element value stands for window area number,0xff means invalid channel

Client SDK Instructions

**DD\_LOG\_INFO**

struct of log information

struct \_dd\_log\_info\_{ unsigned long majorType; unsigned long minorType; unsigned long time; unsigned long IP;

char name [36]; [DD\_TIME](#_page_635_0) localTime; unsigned long infoLen; char info[1024]; }DD\_LOG\_INFO, \*LP\_DD\_LOG\_INFO;

**Members**

*majorType*

major type, refer to DD\_LOG\_CONTENT:

**Type**

DD\_LOG\_CONTENT\_SYSTEM\_CTRL DD\_LOG\_CONTENT\_CONFIG DD\_LOG\_CONTENT\_PLAYBACK DD\_LOG\_CONTENT\_BACKUP DD\_LOG\_CONTENT\_SEARCH DD\_LOG\_CONTENT\_VIEW\_INFO DD\_LOG\_CONTENT\_EVENT\_INFO

DD\_LOG\_CONTENT\_ERROR\_INFO

**Value**

0x00000001 0x00000002 0x00000004 0x00000008 0x00000010 0x00000020 0x00000040

0x00000080

*minorType*

minor type, refer to DD\_LOG\_TYPE:

**DD\_LOG\_TYPE\_SYSTEM\_CTRL**

**Type**

DD\_LOG\_TYPE\_BOOT

**0x01000000**

**description**

boot system

DD\_LOG\_TYPE\_SHUTDOWN

DD\_LOG\_TYPE\_REBOOT

DD\_LOG\_TYPE\_FORMAT\_SUCC

DD\_LOG\_TYPE\_FORMAT\_FAIL

DD\_LOG\_TYPE\_UPGRADE\_SUCC

DD\_LOG\_TYPE\_UPGRADE\_FAIL DD\_LOG\_TYPE\_CLEAR\_ALARM DD\_LOG\_TYPE\_OPEN\_ALARM

DD\_LOG\_TYPE\_MANUAL\_START

DD\_LOG\_TYPE\_MANUAL\_STOP

DD\_LOG\_TYPE\_PTZ\_ENTER

DD\_LOG\_TYPE\_PTZ\_CTRL

DD\_LOG\_TYPE\_PTZ\_EXIT

DD\_LOG\_TYPE\_AUDIO\_CH\_CHANGE

DD\_LOG\_TYPE\_VOLUME\_ADJUST

DD\_LOG\_TYPE\_MUTE\_ENABLE

DD\_LOG\_TYPE\_MUTE\_DISENABLE

DD\_LOG\_TYPE\_DWELL\_ENABLE

DD\_LOG\_TYPE\_DWELL\_DISENABLE

DD\_LOG\_TYPE\_LOG\_IN DD\_LOG\_TYPE\_LOG\_OFF

DD\_LOG\_TYPE\_CHANGE\_TIME

shutdown system

reboot system

format disc successfully

formatting disc fail

upgrade successfully

upgrade fail clear alarm open alarm

open manual record

stop manual record

start PTZ control

PTZ operation

exit PTZ control

chang audio channel

adjust volume

enable mute

disenable mute

enable dwell

disenable dwell

login logout

change system time

DD\_LOG\_TYPE\_MANUAL\_SNAP\_SUCC

DD\_LOG\_TYPE\_MANUAL\_SNAP\_FAIL

**DD\_LOG\_TYPE\_CONFIG**

DD\_LOG\_TYPE\_CHGE\_VIDEO\_FORMAT

DD\_LOG\_TYPE\_CHGE\_VGA\_RESOLUTION

DD\_LOG\_TYPE\_CHGE\_LANGUAGE

DD\_LOG\_TYPE\_CHGE\_NET\_USER\_NUM

DD\_LOG\_TYPE\_CHGE\_TIME\_ZONE

DD\_LOG\_TYPE\_NTP\_MANUAL

DD\_LOG\_TYPE\_NTP\_ON

DD\_LOG\_TYPE\_NTP\_OFF

DD\_LOG\_TYPE\_CHGE\_NTP\_SERVER

DD\_LOG\_TYPE\_CHGE\_DST

manual capture succeed

manual capture fail

**0x02000000**

change video format

change VGA resolution

change language

change network user number

change time zone

manual network time check

enable automatic network time check

disenable automatic network time check

change network time server address

change daylight saving time setting

DD\_LOG\_TYPE\_PASSWD\_ON

DD\_LOG\_TYPE\_PASSWD\_OFF

DD\_LOG\_TYPE\_CHGE\_CAM\_NAME

DD\_LOG\_TYPE\_MODIFY\_COLOR

DD\_LOG\_TYPE\_CHGE\_HOST\_MONITOR

DD\_LOG\_TYPE\_CHGE\_SPOT

DD\_LOG\_TYPE\_CHGE\_OSD

DD\_LOG\_TYPE\_CHGE\_LOCAL\_ENCODE

enable operation password

disappear operation password

change channel name

modify color

change host monitor image setting

change auxiliary output image setting

change character overlap setting

change encoding parameter of record stream

change DD\_LOG\_TYPE\_CHGE\_REC\_VIDEO\_SWITCH record video

switch setting

change DD\_LOG\_TYPE\_CHGE\_REC\_AUDIO\_SWITCHrecord audio

switch setting

DD\_LOG\_TYPE\_CHGE\_REC\_REDU\_SWITCH

DD\_LOG\_TYPE\_CHGE\_REC\_PRE\_TIME

change redundant record switch setting

change the time before record

DD\_LOG\_TYPE\_CHGE\_REC\_POST\_TIME

DD\_LOG\_TYPE\_CHGE\_REC\_HOLD\_TIME

DD\_LOG\_TYPE\_CHGE\_SCH\_SCHEDULE

DD\_LOG\_TYPE\_CHGE\_SCH\_MOTION

DD\_LOG\_TYPE\_CHGE\_SCH\_ALARM

DD\_LOG\_TYPE\_CHGE\_SENSOR\_SWITCH

DD\_LOG\_TYPE\_CHGE\_SENSOR\_TYPE

DD\_LOG\_TYPE\_CHGE\_SENSOR\_TRIGGER

DD\_LOG\_TYPE\_CHGE\_SENSOR\_SCH

DD\_LOG\_TYPE\_CHGE\_MOTION\_SWITCH

DD\_LOG\_TYPE\_CHGE\_MOTION\_SENS

change the time after record

change record data expiry time

change the plan of regular record

change motion detection record schedule

change alarm record schedule

change alarm input switch setting

change alarm input sensor type

change alarm input semsor trigger setting

change alarm input detection schedule

change motion detection switch setting

change motion

DD\_LOG\_TYPE\_CHGE\_MOTION\_AREA

DD\_LOG\_TYPE\_CHGE\_MOTION\_TRIGGER

DD\_LOG\_TYPE\_CHGE\_MOTION\_SCH

DD\_LOG\_TYPE\_CHGE\_VL\_TRIGGER

DD\_LOG\_TYPE\_CHGE\_RELAY\_SWITCH

DD\_LOG\_TYPE\_CHGE\_RELAY\_SCH

DD\_LOG\_TYPE\_BUZZER\_ON

DD\_LOG\_TYPE\_BUZZER\_OFF

DD\_LOG\_TYPE\_CHGE\_BUZZER\_SCH

DD\_LOG\_TYPE\_CHGE\_HTTP\_PORT

DD\_LOG\_TYPE\_CHGE\_SER\_PORT

DD\_LOG\_TYPE\_CHGE\_IP

detection sensitivity

change motion detection area setting

change motion detection process mode

change motion detection schedule

change video lost process mode setting

change alarm output relay setting

change alarm output schedule

enable buzzer alarm

disenable buzzer alarm

change buzzer alarm schedule

modify HTTP server port

modify network server port

change network IP

DD\_LOG\_TYPE\_DHCP\_SUCC

DD\_LOG\_TYPE\_DHCP\_FAIL

DD\_LOG\_TYPE\_CHGE\_PPPOE DD\_LOG\_TYPE\_CHGE\_DDNS

DD\_LOG\_TYPE\_NET\_STREAM\_CFG

DD\_LOG\_TYPE\_CHGE\_SERIAL

DD\_LOG\_TYPE\_PRESET\_MODIFY

DD\_LOG\_TYPE\_CRUISE\_MODIFY

DD\_LOG\_TYPE\_TRACK\_MODIFY DD\_LOG\_TYPE\_USER\_ADD

DD\_LOG\_TYPE\_USER\_MODIFY

DD\_LOG\_TYPE\_USER\_DELETE

DD\_LOG\_TYPE\_CHANGE\_PASSWD

DD\_LOG\_TYPE\_LOAD\_DEFAULT

DD\_LOG\_TYPE\_IMPORT\_CONFIG

DD\_LOG\_TYPE\_EXPORT\_CONFIG

DD\_LOG\_TYPE\_CHGE\_IMAGE\_MASK

obtain DHCP automatically succeed

obtain DHCP automatically fail

set PPPoE set DDNS

change network stream edcoding setting

change PTZ serial port setting

modify preset point

modify cruise line

modify track add users

modify user authority

delete user

modify user password

recover default configuration

import configuration

export configuration

image shield

DD\_LOG\_TYPE\_RECYCLE\_REC\_ON

DD\_LOG\_TYPE\_RECYCLE\_REC\_OFF

DD\_LOG\_TYPE\_CHGE\_DISK\_ALARM

DD\_LOG\_TYPE\_CHGE\_SEND\_EMAIL

DD\_LOG\_TYPE\_CHGE\_RECV\_EMAIL

DD\_LOG\_TYPE\_CHGE\_SNAP\_SETTING

**DD\_LOG\_TYPE\_PLAYBACK**

DD\_LOG\_TYPE\_PLAYBACK\_PLAY DD\_LOG\_TYPE\_PLAYBACK\_PAUSE DD\_LOG\_TYPE\_PLAYBACK\_RESUME DD\_LOG\_TYPE\_PLAYBACK\_FF DD\_LOG\_TYPE\_PLAYBACK\_REW DD\_LOG\_TYPE\_PLAYBACK\_STOP

DD\_LOG\_TYPE\_PLAYBACK\_NEXT\_SECTION

DD\_LOG\_TYPE\_PLAYBACK\_PREV\_SECTION

**DD\_LOG\_TYPE\_BACKUP**

DD\_LOG\_TYPE\_BACKUP\_START

DD\_LOG\_TYPE\_BACKUP\_COMPLETE

DD\_LOG\_TYPE\_BACKUP\_CANCEL

DD\_LOG\_TYPE\_BACKUP\_FAIL

**DD\_LOG\_TYPE\_SEARCH**

enable loop record

close loop record

change disc alarm space

set Email sender information

set Email receiver information

change capture configuration

**0x03000000**

play pause

resume play fast forward rewind

stop

play next section

play previous section

**0x04000000**

start to backup

backup is completed

cancel backup

backup fails

**0x05000000**

DD\_LOG\_TYPE\_SEARCH\_TIME

DD\_LOG\_TYPE\_SEARCH\_EVENT

DD\_LOG\_TYPE\_SEARCH\_FILE\_MAN

DD\_LOG\_TYPE\_SEARCH\_PICTURE

DD\_LOG\_TYPE\_DELETE\_FILE DD\_LOG\_TYPE\_LOCK\_FILE DD\_LOG\_TYPE\_UNLOCK\_FILE DD\_LOG\_TYPE\_DELETE\_PICTURE DD\_LOG\_TYPE\_LOCK\_PICTURE

DD\_LOG\_TYPE\_UNLOCK\_PICTURE

**DD\_LOG\_TYPE\_EVENT\_INFO**

DD\_LOG\_TYPE\_SENSOR\_START

DD\_LOG\_TYPE\_SENSOR\_END

DD\_LOG\_TYPE\_MOTION\_START

DD\_LOG\_TYPE\_MOTION\_END

DD\_LOG\_TYPE\_VLOSS\_START

DD\_LOG\_TYPE\_VLOSS\_END

DD\_LOG\_TYPE\_SHELTER\_START

DD\_LOG\_TYPE\_SHELTER\_END

**DD\_LOG\_TYPE\_BEHAVIOR\_INFO**

search by time

search by event

search file management

search picture

delete file lock file unlock file delete picture lock picture

unlock picture

**0x07000000**

start sensor alarm

sensor alarm ends

motion detection starts

motion detection ends

video loss start

video loss ends

video shelter starts

video shelter ends

**0x08000000**

DD\_LOG\_TYPE\_ENTER\_AREA DD\_LOG\_TYPE\_EXIT\_AREA DD\_LOG\_TYPE\_INTRUSION DD\_LOG\_TYPE\_LOITER

DD\_LOG\_TYPE\_LEFT\_TAKE

DD\_LOG\_TYPE\_PARKING DD\_LOG\_TYPE\_RUN

DD\_LOG\_TYPE\_HIGH\_DENSITY

**DD\_LOG\_TYPE\_ERROR\_INFO**

DD\_LOG\_TYPE\_IP\_CONFLICT

DD\_LOG\_TYPE\_NETWORK\_ERR

DD\_LOG\_TYPE\_DDNS\_ERR

DD\_LOG\_TYPE\_DISK\_IO\_ERR

DD\_LOG\_TYPE\_UNKNOWN\_OFF

DD\_LOG\_TYPE\_UNKNOWN\_ERR

enter area exit area intrusion loiter

across left cordon

parking run

high density behaviour

**0x09000000**

network IP conflict

network exception

DDNS error

disc read-write error

electricity outage exception

unknown error

*time*

log occurrence time *IP*

user IP *name [36]*

user name *localTime*

local time, later fill in *infoLen*

length of log information *info[1024]*

length of the log information

Client SDK Instructions

**DD\_MOTION\_AREA**

struct of motion area

struct \_dd\_motion\_area\_{

unsigned long sensitivity; unsigned short widthNum; unsigned short hightNum; unsigned char area

[DD\_MAX\_MOTION\_AREA\_HIGHT\_NUM] [DD\_MAX\_MOTION\_AREA\_WIDTH\_NUM];

}DD\_MOTION\_AREA;

**Members**

*sensitivity*

sensitivity(0-7),high number means high sensitivity *widthNum*

width grid number of area *hightNum*

height grid number of area

*area [DD\_MAX\_MOTION\_AREA\_HIGHT\_NUM] [DD\_MAX\_MOTION\_AREA\_WIDTH\_NUM]*

grid mask data of area,compatible 1920x1080,each size is 8X8

Client SDK Instructions

**DD\_MOTION\_CONFIG**

struct of motion configuration

struct \_dd\_motion\_config\_{

unsigned long iSize; unsigned char enable; unsigned char recv; unsigned short holdTime; [DD\_MOTION\_AREA](#_page_598_0) area; }DD\_MOTION\_CONFIG;

**Members**

*iSize*

size of the struct *enable*

whether enable motion detection *recv*

reserved byte *holdTime*

delay time *area*

area setting

Client SDK Instructions

**DD\_NETWORK\_ADVANCE\_CONFIG**

struct of network advanced configuration

struct \_dd\_network\_advance\_config\_{ unsigned long iSize; unsigned char bMessagePort; unsigned char bAlarmPort; unsigned char bMultiCastIP; unsigned char bMTUByteNum; unsigned short httpPort; unsigned short datePort; unsigned short messagePort; unsigned short alarmPort;

unsigned short maxOnlineUserNum; unsigned short OnlineUserNum; unsigned long multiCastIP; unsigned long mtuByteNum; }DD\_NETWORK\_ADVANCE\_CONFIG;

**Members**

*iSize*

size of the struct *bMessagePort*

whether support message port(read only) *bAlarmPort*

whether support alarm port(read only) *bMultiCastIP*

whether support multicast address(read only) *bMTUByteNum*

whether support MTU byte number(read only) *httpPort*

HTTP port *datePort*

data port *messagePort*

message command port

*alarmPort alarm port*

*maxOnlineUserNum*

*supportable maximum online user number(read only) OnlineUserNum*

*number of online users multiCastIP*

*multicast address mtuByteNum*

*MTU byte number*

Client SDK Instructions

**DD\_NETWORK\_IP\_CONFIG**

struct of network IP configuration

struct \_dd\_network\_ip\_config\_{ unsigned long iSize; unsigned long useDHCP; unsigned long IP; unsigned long subnetMask; unsigned long gateway;

unsigned long preferredDNS; unsigned long alternateDNS; unsigned long usePPPoE;

char account[DD\_MAX\_PPPOE\_ACCOUNT\_BUF\_LEN]; char password[DD\_MAX\_PASSWORD\_BUF\_LEN]; }DD\_NETWORK\_IP\_CONFIG;

**Members**

*iSize*

size of the struct *useDHCP*

whether enable DHCP *IP*

network IP *subnetMask*

subnet mask *gateway*

gateway *preferredDNS*

host DNS *alternateDNS*

alternate DNS *usePPPoE*

whether enable PPPoE *account[DD\_MAX\_PPPOE\_ACCOUNT\_BUF\_LEN]/i>*

*PPPoE account password[DD\_MAX\_PASSWORD\_BUF\_LEN]*

*PPPoE password*

Client SDK Instructions

**DD\_POSITION**

struct of position

struct \_dd\_position\_{ unsigned short x; unsigned short y; }DD\_POSITION;

**Members**

*x*

abscissa *y*

ordinate

Client SDK Instructions

**DD\_PTZ\_CONFIG**

struct of PTZ configuration

struct \_dd\_ptz\_config\_{ unsigned long

unsigned char unsigned char unsigned char unsigned char unsigned long [DD\_SERIAL\_CONFIG](#_page_630_0) }DD\_PTZ\_CONFIG;

iSize; enable; address; recv1; recv2; protocol; serial;

**Members**

*iSize*

size of the struct *enable*

whether enable PTZ function *address*

address *recv1*

reserved byte *recv2*

reserved byte *protocol*

protocol,refer to PROTOCOL\_TYPE:

**Type** **Value**

PROTOCOL\_NULL 0 PROTOCOL\_PELCOP 1 PROTOCOL\_PELCOD 2 PROTOCOL\_LILIN 3 PROTOCOL\_MINKING 4

PROTOCOL\_NEON 5 PROTOCOL\_STAR 6 PROTOCOL\_VIDO 7 PROTOCOL\_DSCP 8 PROTOCOL\_VISCA 9 PROTOCOL\_SAMSUNG 10 PROTOCOL\_RM110 11 PROTOCOL\_HY 12

*serial*

serial port

Client SDK Instructions

**DD\_PTZ\_PRESET\_CONFIG**

struct of PTZ preset configuration

struct \_dd\_ptz\_preset\_config\_{ unsigned long iSize;

unsigned char enablePreset [DD\_MAX\_PRESET\_NUM];

}DD\_PTZ\_PRESET\_CONFIG;

**Members**

*iSize*

size of the struct

*enablePreset [DD\_MAX\_PRESET\_NUM]* whether enable preset point

Client SDK Instructions

**DD\_PTZ\_PROTOCOL\_INFO**

struct of PTZ protocol information

struct \_dd\_ptz\_protocol\_info unsigned long

unsigned long char

}DD\_PTZ\_PROTOCOL\_INFO;

{ protocolID;

pportproperty; ProtocolName[64];

**Members**

*protocolID* protocol type ID

*pportproperty*

other attribute's MASK except ID,baud rate, such as whether support some special attribute track etc.

*ProtocolName[64]* protocal name

Client SDK Instructions

**DD\_RECORD\_CONFIG**

struct of record configuration

struct \_dd\_record\_config\_{ unsigned long iSize; unsigned char bOnlyVideo; unsigned char bWithAudio;

unsigned char bindAudioChannel; unsigned char bRedundancy; unsigned short preAlarmTime; unsigned short postAlarmTime; unsigned short expired;

unsigned short recv; }DD\_RECORD\_CONFIG;

**Members**

*iSize*

size of the struct *bOnlyVideo*

transcript video(only video) *bWithAudio*

transcript audio(based on transcript video) *bindAudioChannel*

corresponding audio channel(may different from video channel number)

*bRedundancy*

whether redundant record *preAlarmTime*

time of record before alarm *postAlarmTime*

time of record after alarm *expired*

record expiry time *recv*

reserved byte

Client SDK Instructions

**DD\_RECORD\_CONFIG\_MASK**

struct of record configuration mask

struct \_dd\_record\_config\_mask\_{ unsigned long iSize;

unsigned char bindAudioChannel; unsigned char bRedundancy; unsigned char recv1;

unsigned char recv2;

unsigned short minPreAlarmTime; unsigned short maxPreAlarmTime; unsigned short minPostAlarmTime; unsigned short maxPostAlarmTime; unsigned short minExpired; unsigned short maxExpired; }DD\_RECORD\_CONFIG\_MASK;

**Members**

*iSize*

size of the struct *bindAudioChannel*

whether support to bind audio and video channel *bRedundancy*

whether support redundant record *recv1*

reserved byte *recv2*

reserved byte *minPreAlarmTime*

the minimum time of record before alarm *maxPreAlarmTime*

the maximum time of record before alarm *minPostAlarmTime*

the minimum time of record after alarm *maxPostAlarmTime*

the maximum time of record after alarm

*minExpired*

the minimum record data expiry time *maxExpired*

the maximum record data expiry time

Client SDK Instructions

**DD\_RECORD\_LOG**

struct of record log information

struct\_dd\_record\_log\_{ unsigned char bLocked; unsigned char bUnofficial; unsigned char enableCard; unsigned char recv1; unsigned short diskIndex; unsigned short fileIndex; unsigned short logIndex; unsigned short recv2; unsigned short deviceID; unsigned short cameraID; unsigned long channel; unsigned long type; unsigned long size;

DD\_TIME startTime; DD\_TIME endTime; char cardNo[32]; }DD\_RECORD\_LOG, \*LP\_DD\_RECORD\_LOG;

**Members**

*bLocked*

0 means unlocked,1 means locked *bUnofficial*

0 means official record,1 means unofficial record(overlap record after modifying time)

*enableCard*

whether enable card *recv1*

reserved byte *diskIndex*

disk number *fileIndex*

file index *logIndex*

log index *recv2*

reserved byte *deviceID*

device ID *cameraID*

camera ID *channel*

virtual channel number *type*

record type *size*

size of the record data *startTime*

start time *endTime*

end time *cardNo[32]*

card number

Client SDK Instructions

**DD\_RELAY\_CONFIG**

struct of delay configuration

struct \_dd\_relay\_config\_{ unsigned char enable; unsigned char recv; unsigned short holdTime;

char name [DD\_MAX\_NAME\_BUF\_LEN]; }DD\_RELAY\_CONFIG;

**Members**

*enable*

alarm output device enable switch *recv*

reserved byte *holdTime*

delay time

*name [DD\_MAX\_NAME\_BUF\_LEN]* device name

Client SDK Instructions

**DD\_SENSOR\_CONFIG**

struct of sensor configuration

struct \_dd\_sensor\_config\_{ unsigned long iSize;

unsigned char enable; unsigned char bNO; unsigned short holdTime;

char name [DD\_MAX\_NAME\_BUF\_LEN]; }DD\_SENSOR\_CONFIG;

**Members**

*iSize*

size of the struct *enable*

whether enable detection *bNO*

device type:normal open or normal close *holdTime*

delay time

*name [DD\_MAX\_NAME\_BUF\_LEN]* device name

Client SDK Instructions

**DD\_SERIAL\_CONFIG**

struct of serial configuration

struct \_dd\_serial\_config\_{ unsigned long baudRate; unsigned long dataBit; unsigned long stopBit; unsigned long parity;

unsigned long dataFlowControl; }DD\_SERIAL\_CONFIG;

**Members**

*baudRate*

baud rate,refer to the list below:

**Type** **Value**

SBR\_110 0 SBR\_300 1 SBR\_600 2 SBR\_1200 3 SBR\_2400 4 SBR\_4800 5 SBR\_9600 6 SBR\_19200 7 SBR\_38400 8 SBR\_57600 9 SBR\_115200 10 SBR\_230400 11 SBR\_460800 12 SBR\_921600 13

*dataBit*

data bit,refer to the list below:

**Type** **Value**

DATABITS7 7 DATABITS8 8

*stopBit*

stop bit,refer to the list below:

**Type** **Value**

STOPBITS1 2 STOPBITSONEHALF 3 STOPBITS2 4

*parity*

parity check bit,refer to the list below:

**Type**

PARITYEVEN PARITYODD PARITYMARK PARITYSPACE

PARITYNONE

**Value** **Description**

'E' even parity check 'O' odd parity check 'M' mark parity check 'S' space parity check

'N' no parity check

*dataFlowControl*

data stream control

Client SDK Instructions

**DD\_SMTP\_CONFIG**

struct of SMTP configuration

struct \_dd\_smtp\_config\_{

unsigned long iSize; unsigned short port; unsigned short enableSSL;

char server [DD\_MAX\_URL\_BUF\_LEN]; char sendAddress

[DD\_MAX\_URL\_BUF\_LEN];

char password [DD\_MAX\_PASSWORD\_BUF\_LEN];

unsigned long enableRecvAddrNum; £© char receiveAddress

[DD\_MAX\_EMAIL\_RECEIVE\_ADDR\_NUM][DD\_MAX\_URL\_BUF\_LEN]; }DD\_SMTP\_CONFIG;

**Members**

*iSize*

size of the struct *port*

SMTP server port *enableSSL*

whether enable SSL check *server [DD\_MAX\_URL\_BUF\_LEN]*

send server address

*sendAddress [DD\_MAX\_URL\_BUF\_LEN]* send SMTP address

*password [DD\_MAX\_PASSWORD\_BUF\_LEN]* password

*enableRecvAddrNum*

available address number for receiving(read only) *receiveAddress [DD\_MAX\_EMAIL\_RECEIVE\_ADDR\_NUM] [DD\_MAX\_URL\_BUF\_LEN]*

list of acception address

Client SDK Instructions

**DD\_TIME**

struct of system time setting information.

struct \_dd\_time\_{

unsigned char second; unsigned char minute; unsigned char hour; unsigned char wday; unsigned char mday; unsigned char month; unsigned short year; }DD\_TIME, \*LP\_DD\_TIME;

**Members**

*second*

seconds after minute,range (0-59). *minute*

minutes after hour,range (0-59). *hour*

hours since midnight,range (0-23). *wday*

day of week ,range(0-6; Sunday=0). *mday*

day of month,range (1-31). *month*

month range(0-11; January=0). *year*

year (current year minus 1900).

Client SDK Instructions

**DD\_TRIGGER\_ALARM\_OUT**

struct of triggering alarm

struct \_dd\_trigger\_alarm\_out\_{ unsigned char toBuzzer;

unsigned char ShowFullScreen; unsigned char sendEmail;

unsigned char toUploadToAlarmCentre; unsigned long toAlarmOut; }DD\_TRIGGER\_ALARM\_OUT;

**Members**

*toBuzzer*

trigger buzzer alarm *ShowFullScreen*

trigger full screen alarm(no trigger when channel number is 0xff)

*sendEmail* send email

*toUploadToAlarmCentre* upload to alarm center

*toAlarmOut*

alarm output(bit matches output device)

Client SDK Instructions

**DD\_TRIGGER\_PTZ**

struct of triggering PTZ

struct \_dd\_trigger\_ptz\_{ unsigned char

unsigned char unsigned char unsigned char }DD\_TRIGGER\_PTZ;

toPTZType; toIndex; backIndex; recv;

**Members**

*toPTZType*

linkage typeŁ¬refer to the list below:

**Type** **Value**

DD\_PTZ\_TYPE\_PRESET 1 DD\_PTZ\_TYPE\_CRUISE 2 DD\_PTZ\_TYPE\_TRACE 3

*toIndex*

linkage number(preset point, cruise line, track) *backIndex*

linkage returned number(preset point, cruise line, track)

*recv*

reserved byte

Client SDK Instructions

**DD\_TRIGGER\_RECORD**

struct of triggering video record

struct \_dd\_trigger\_record\_{ unsigned char snapCH

[DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; unsigned char recordCH

[DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]; }DD\_TRIGGER\_RECORD;

**Members**

*snapCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* trigger capture

*recordCH [DD\_MAX\_CAMERA\_NUM\_BYTE\_LEN]* trigger record

Client SDK Instructions

**DD\_VIDEO\_COLOR**

struct of video color

struct \_dd\_video\_color\_{ unsigned long startTime; unsigned char brightness; unsigned char hue; unsigned char saturation; unsigned char contrast; }DD\_VIDEO\_COLOR;

**Members**

*startTime*

start time of the color(relative time in one day) *brightness*

brightness ,range 0-255 *hue*

hue ,range 0-255 *saturation*

saturation ,range 0-255 *contrast*

contrast ,range 0-255

Client SDK Instructions

**DD\_VIDEO\_COLOR\_CONFIG**

struct of video color configuration

struct \_dd\_video\_color\_config\_{ unsigned long iSize; unsigned long usedNum;

[DD\_VIDEO\_COLOR](#_page_643_0) videoColor[DD\_MAX\_COLOR\_CFG\_NUM]; }DD\_VIDEO\_COLOR\_CONFIG;

**Members**

*iSize*

size of the struct *usedNum*

scheme number in use, default value is 1 *videoColor[DD\_MAX\_COLOR\_CFG\_NUM]*

color scheme

Client SDK Instructions

**DD\_VIDEO\_OSD\_CONFIG**

struct of video OSD configuration

struct \_dd\_video\_osd\_config\_{ unsigned long iSize;

unsigned char enableCameraName; unsigned char enableTimeStamp; unsigned char enableTimeStampWithWeek; unsigned char enableDefineText; [DD\_POSITION](#_page_608_0) cameraName;

[DD\_POSITION](#_page_608_0) timeStamp; [DD\_POSITION](#_page_608_0) defineText;

char text [DD\_MAX\_TEXT\_BUF\_LEN]; struct

{

unsigned long enable; [DD\_AREA](#_page_537_0) area;

}cover[DD\_MAX\_VIDEO\_COVER\_NUM]; }DD\_VIDEO\_OSD\_CONFIG;

**Members**

*iSize*

size of the struct *enableCameraName*

overlap channel name *enableTimeStamp*

overlap timestamp *enableTimeStampWithWeek*

timestamp with week *enableDefineText*

overlap self-defined text *cameraName*

channel name position,range from (0,0) to (100,100) *timeStamp*

timestamp position,range from (0,0) to (100,100) *defineText*

self-defined text position,range from (0,0) to (100,100)

*text [DD\_MAX\_TEXT\_BUF\_LEN]* self-defined text

*enable*

area overlap is valid or not *area*

area parameter *cover[DD\_MAX\_VIDEO\_COVER\_NUM]*

covered area parameter

Client SDK Instructions

**DD\_WEEK\_SCHEDULE**

struct of week schedule

struct \_dd\_week\_schedule\_{ DD\_DATE\_SCHEDULE week[7]; }DD\_WEEK\_SCHEDULE;

**Members**

*week[7]*

week schedule structure,7 stands for each day's schedule in a week of 7 days

Client SDK Instructions

**DEC\_ADVANCE\_NETWORK**

struct of decoder advanced network configuration

struct \_dec\_advance\_network{ unsigned long iSize; unsigned char TimeZone; unsigned char hour; unsigned char min; unsigned char sec; unsigned char mday; unsigned char month; unsigned short year;

bool enableFlag; char name [132]; int NTP\_Port;

int syncInterval; }DEC\_ADVANCE\_NETWORK;

**Members**

*iSize*

size of the struct *TimeZone*

time zone, refer to DD\_TIME\_ZONE\_NAME:

**Type** **Value**

DD\_TIME\_ZONE\_GMT\_D12 0 DD\_TIME\_ZONE\_GMT\_D11 1 DD\_TIME\_ZONE\_GMT\_D10 2 DD\_TIME\_ZONE\_GMT\_D9 3 DD\_TIME\_ZONE\_GMT\_D8 4 DD\_TIME\_ZONE\_GMT\_D7 5 DD\_TIME\_ZONE\_GMT\_D6 6 DD\_TIME\_ZONE\_GMT\_D5 7

DD\_TIME\_ZONE\_GMT\_D4\_30 8 DD\_TIME\_ZONE\_GMT\_D4 9 DD\_TIME\_ZONE\_GMT\_D3\_30 10 DD\_TIME\_ZONE\_GMT\_D3 11 DD\_TIME\_ZONE\_GMT\_D2 12 DD\_TIME\_ZONE\_GMT\_D1 13 DD\_TIME\_ZONE\_GMT 14 DD\_TIME\_ZONE\_GMT\_A1 15 DD\_TIME\_ZONE\_GMT\_A2 16 DD\_TIME\_ZONE\_GMT\_A3 17 DD\_TIME\_ZONE\_GMT\_A3\_30 18 DD\_TIME\_ZONE\_GMT\_A4 19 DD\_TIME\_ZONE\_GMT\_A4\_30 20 DD\_TIME\_ZONE\_GMT\_A5 21 DD\_TIME\_ZONE\_GMT\_A5\_30 22 DD\_TIME\_ZONE\_GMT\_A5\_45 23 DD\_TIME\_ZONE\_GMT\_A6 24 DD\_TIME\_ZONE\_GMT\_A6\_30 25 DD\_TIME\_ZONE\_GMT\_A7 26 DD\_TIME\_ZONE\_GMT\_A8 27 DD\_TIME\_ZONE\_GMT\_A9 28 DD\_TIME\_ZONE\_GMT\_A9\_30 29 DD\_TIME\_ZONE\_GMT\_A10 30 DD\_TIME\_ZONE\_GMT\_A11 31 DD\_TIME\_ZONE\_GMT\_A12 32 DD\_TIME\_ZONE\_GMT\_A13 33

*hour* hour

*min* minute

*sec*

second *mday*

which day in a week *month*

which month in a year *year*

a particular year,2008-2025 *enableFlag*

NTP enable flag *name [132]*

NTP server address *NTP\_Port*

NTP server port *syncInterval*

synchronous time interval,in hour

Client SDK Instructions

**DEC\_DATE\_SCHEDULE**

struct of decoder date schedule

struct \_dec\_date\_schedule\_{

unsigned long long hour [24]; }DEC\_DATE\_SCHEDULE;

**Members**

*hour [24]*

time format,24 means in 24-hours time system, each bit of unsigned long long stands for each minute's status

Client SDK Instructions

**DEC\_DEVICE\_CONFIG**

struct of decoder configuration

struct \_dec\_device\_config{ unsigned long

char [DEC\_MAX\_NAME\_LEN];

unsigned long unsigned long unsigned long unsigned long char

[DEC\_MAX\_VERSION\_BUF\_LEN]; char

[DEC\_MAX\_VERSION\_BUF\_LEN]; char

[DEC\_MAX\_VERSION\_BUF\_LEN]; }DEC\_DEVICE\_CONFIG;

iSize; deviceName

channelNum; productID; productSubID; softVersion;

mcuVersion

kernelVersion

hardwareVersion

**Members**

*iSize*

size of the struct

*deviceName [DEC\_MAX\_NAME\_LEN]*

device name(notice double byte character) *channelNum*

sum of decoder channels *productID*

product ID *productSubID*

product sub ID *softVersion*

soft version

*mcuVersion [DEC\_MAX\_VERSION\_BUF\_LEN]* MCU version

*kernelVersion [DEC\_MAX\_VERSION\_BUF\_LEN]*

kernel version

*hardwareVersion [DEC\_MAX\_VERSION\_BUF\_LEN]* hardware version

Client SDK Instructions

**DEC\_NETWORK\_CONFIG**

struct of decoder network configuration

struct \_dec\_network\_config{ unsigned long

unsigned long unsigned long unsigned long unsigned short unsigned short char

unsigned long unsigned long unsigned long unsigned long }DEC\_NETWORK\_CONFIG;

iSize; IP;

subnetMask; gateway;

httpPort; decoderPort; MAC [8]; multiCastIP; bDHCP;

dns1; dns2;

**Members**

*iSize*

size of the struct *IP*

network address *subnetMask*

sub net mask *gateway*

gateway *httpPort*

HTTP port *decoderPort*

decoder port *MAC [8]*

binded MAC *multiCastIP*

multicast address

*dns1* DNS1

*dns2* DNS2

Client SDK Instructions

**DEC\_OTHER\_ALARM**

other alarm struct of decoder

struct \_dec\_other\_alarm\_{ unsigned long iSize;

unsigned long toBuzzerForIPConflict; unsigned long toAlarmOutForDisconnect; unsigned long toBuzzerForDisconnect; unsigned long toAlarmOutForIPConflict; }DEC\_OTHER\_ALARM;

**Members**

*iSize*

size of the struct *toBuzzerForIPConflict*

IP conflict triggers buzzer *toAlarmOutForDisconnect*

IP conflict alarm output(bit matches output device) *toBuzzerForDisconnect*

network disconnection triggers buzzer *toAlarmOutForIPConflict*

network disconnection alarm output(bit matches output device)

Client SDK Instructions

**DEC\_SENSOR\_SETUP**

struct of decoder sensor setting

struct \_dec\_sensor\_setup\_{ unsigned long iSize;

unsigned char enable; unsigned char bNO; unsigned short holdTime;

char name [DEC\_MAX\_BUF\_LEN]; unsigned long toBuzzer;

unsigned long toAlarmOut; }DEC\_SENSOR\_SETUP;

**Members**

*iSize*

size of the struct *enable*

whether enable detection *bNO*

device type:normal open or normal close *holdTime*

delay time

*name [DEC\_MAX\_BUF\_LEN]* device name

*toBuzzer*

trigger buzzer alarm *toAlarmOut*

alarm output(bit matches output device)

Client SDK Instructions

**DEC\_WEEK\_SCHEDULE**

struct of decoder week schedule

struct \_dec\_week\_schedule\_{ [DEC\_DATE\_SCHEDULE](#_page_656_0) week[7]; }DEC\_WEEK\_SCHEDULE;

**Members**

*week[7]*

week schedule structure,7 stands for 7 days' schedule in a week

Client SDK Instructions

**NET\_SDK\_ALARMINFO**

struct of alarm information

struct \_net\_sdk\_alarminfo{ DWORD dwAlarmType; DWORD dwSensorIn; DWORD dwChannel; DWORD dwDisk; }NET\_SDK\_ALARMINFO;

**Members**

*dwAlarmType*

alarm type,refer to the following list:

**Type** NET\_SDK\_N9000\_ALARM\_TYPE\_MOTION NET\_SDK\_N9000\_ALARM\_TYPE\_SENSOR NET\_SDK\_N9000\_ALARM\_TYPE\_VLOSS NET\_SDK\_N9000\_ALARM\_TYPE\_FRONT\_OFFLINE NET\_SDK\_N9000\_ALARM\_TYPE\_OSC NET\_SDK\_N9000\_ALARM\_TYPE\_AVD

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_SECENE

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_CLARITY

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_COLOR NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_TRIPWIRE NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_PERIMETER NET\_SDK\_N9000\_ALARM\_TYPE\_VFD NET\_SDK\_N9000\_ALARM\_TYPE\_CDD NET\_SDK\_N9000\_ALARM\_TYPE\_IPD NET\_SDK\_N9000\_ALARM\_TYPE\_CPC NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH\_FOR\_IPC NET\_SDK\_N9000\_ALARM\_TYPE\_TRAJECT NET\_SDK\_N9000\_ALARM\_TYPE\_VEHICE NET\_SDK\_N9000\_ALARM\_TYPE\_AOIENTRY NET\_SDK\_N9000\_ALARM\_TYPE\_AOILEAVE NET\_SDK\_N9000\_ALARM\_TYPE\_PASSLINE NET\_SDK\_N9000\_ALARM\_TYPE\_IP\_CONFLICT NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_IO\_ERROR NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_FULL NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_SUBHEALTH NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_UNAVAILABLE NET\_SDK\_N9000\_ALARM\_TYPE\_ILLEIGAL\_ACCESS NET\_SDK\_N9000\_ALARM\_TYPE\_NET\_DISCONNECT NET\_SDK\_N9000\_ALARM\_TYPE\_NO\_DISK NET\_SDK\_N9000\_ALARM\_TYPE\_SIGNAL\_SHELTER

**Description** Motiondetection alarm Sensor alarm input

Video loss alarm

Front-end deviceofflinealarm Object Abandoned/Missing al Exception alarm

Exception detection-

Scene change, for IPC only Exception detection-

video blurred, for IPC only Exception detection-

video color cast, for IPC only Tripwire alarm

Intrusion alarm

Face Detection (currently IPC only) Crowddensity detection Intrusion persondetection People counting

Face matchalarm(NVR) Face matchalarm(IPC) Target tracking trajectory license plate(IPC)

Enter the area(IPC) Leave the area(IPC) Tripwirecounting IP address conflict Disk IO error

Full disk

Array sub-health Array unavailable Illegal access

Network disconnect No disk in disk group Signal shelter

NET\_SDK\_N9000\_ALARM\_TYPE\_HDD\_PULL\_OUT Front panel HDDpull out

the alarmtype NET\_SDK\_ALARM\_TYPE(refer to the following list) is not be used now,it is invalid

**Type**

NET\_SDK\_ALARM\_TYPE\_MOTION NET\_SDK\_ALARM\_TYPE\_SENSOR NET\_SDK\_ALARM\_TYPE\_VLOSS NET\_SDK\_ALARM\_TYPE\_SHELTER

NET\_SDK\_ALARM\_TYPE\_DISK\_FULL

**Description**

motion detection sensor alarm single loss shelter alarm

full harddisk

NET\_SDK\_ALARM\_TYPE\_DISK\_UNFORMATTEDdisc unformatted

NET\_SDK\_ALARM\_TYPE\_DISK\_WRITE\_FAIL

NET\_SDK\_ALARM\_TYPE\_EXCEPTION

harddisk read-write error

exception alarm

*dwSensorIn*

sensor alarm input port number *dwChannel*

when alarm is relative to channel,dwChannel means alarm channel *dwDisk*

in disc alarming, it means disc number which alarms

Client SDK Instructions

**NET\_SDK\_ALARMINFO\_EX**

struct of alarm information

struct \_net\_sdk\_alarminfo\_ex{ DWORD dwAlarmType; DWORD dwSensorIn; DWORD dwChannel; DWORD dwDisk;

char sensorName[36];

char alarmTime[20]; char resv[128];

}NET\_SDK\_ALARMINFO\_EX;

**Members**

*dwAlarmType*

alarm type,refer to the following list:

**Type** NET\_SDK\_N9000\_ALARM\_TYPE\_MOTION NET\_SDK\_N9000\_ALARM\_TYPE\_SENSOR NET\_SDK\_N9000\_ALARM\_TYPE\_VLOSS NET\_SDK\_N9000\_ALARM\_TYPE\_FRONT\_OFFLINE NET\_SDK\_N9000\_ALARM\_TYPE\_OSC NET\_SDK\_N9000\_ALARM\_TYPE\_AVD

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_SECENE

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_CLARITY

NET\_SDK\_N9000\_ALARM\_TYPE\_AVD\_COLOR NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_TRIPWIRE NET\_SDK\_N9000\_ALARM\_TYPE\_PEA\_PERIMETER NET\_SDK\_N9000\_ALARM\_TYPE\_VFD NET\_SDK\_N9000\_ALARM\_TYPE\_CDD NET\_SDK\_N9000\_ALARM\_TYPE\_IPD NET\_SDK\_N9000\_ALARM\_TYPE\_CPC NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH NET\_SDK\_N9000\_ALARM\_TYPE\_FACE\_MATCH\_FOR\_IPC NET\_SDK\_N9000\_ALARM\_TYPE\_TRAJECT NET\_SDK\_N9000\_ALARM\_TYPE\_VEHICE NET\_SDK\_N9000\_ALARM\_TYPE\_AOIENTRY NET\_SDK\_N9000\_ALARM\_TYPE\_AOILEAVE NET\_SDK\_N9000\_ALARM\_TYPE\_PASSLINE NET\_SDK\_N9000\_ALARM\_TYPE\_IP\_CONFLICT NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_IO\_ERROR NET\_SDK\_N9000\_ALARM\_TYPE\_DISK\_FULL NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_SUBHEALTH NET\_SDK\_N9000\_ALARM\_TYPE\_RAID\_UNAVAILABLE NET\_SDK\_N9000\_ALARM\_TYPE\_ILLEIGAL\_ACCESS NET\_SDK\_N9000\_ALARM\_TYPE\_NET\_DISCONNECT

**Description** Motiondetection alarm Sensor alarm input

Video loss alarm

Front-end deviceofflinealarm Object Abandoned/Missing al Exception alarm

Exception detection-

Scene change, for IPC only Exception detection-

video blurred, for IPC only Exception detection-

video color cast, for IPC only Tripwire alarm

Intrusion alarm

Face Detection (currently IPC only) Crowddensity detection Intrusion persondetection People counting

Face matchalarm(NVR) Face matchalarm(IPC) Target tracking trajectory license plate(IPC)

Enter the area(IPC) Leave the area(IPC) Tripwirecounting IP address conflict Disk IO error

Full disk

Array sub-health Array unavailable Illegal access

Network disconnect

NET\_SDK\_N9000\_ALARM\_TYPE\_NO\_DISK No disk in disk group NET\_SDK\_N9000\_ALARM\_TYPE\_SIGNAL\_SHELTER Signal shelter NET\_SDK\_N9000\_ALARM\_TYPE\_HDD\_PULL\_OUT Front panel HDDpull out

the alarmtype NET\_SDK\_ALARM\_TYPE(refer to the following list) is not be used now,it is invalid

**Type**

NET\_SDK\_ALARM\_TYPE\_MOTION NET\_SDK\_ALARM\_TYPE\_SENSOR NET\_SDK\_ALARM\_TYPE\_VLOSS NET\_SDK\_ALARM\_TYPE\_SHELTER

NET\_SDK\_ALARM\_TYPE\_DISK\_FULL

**Description**

motion detection sensor alarm single loss shelter alarm

full harddisk

NET\_SDK\_ALARM\_TYPE\_DISK\_UNFORMATTEDdisc unformatted

NET\_SDK\_ALARM\_TYPE\_DISK\_WRITE\_FAIL

NET\_SDK\_ALARM\_TYPE\_EXCEPTION

harddisk read-write error

exception alarm

*dwSensorIn*

sensor alarm input port number *dwChannel*

when alarm is relative to channel,dwChannel means alarm channel *dwDisk*

in disc alarming, it means disc number which alarms *sensorName*

in sensor alarming, it means the alarming sensor's name *alarmTime*

alarm time *resv*

preserve

Client SDK Instructions

**NET\_SDK\_CLIENTINFO**

struct of log information

struct \_net\_sdk\_clientinfo{ LONG lChannel;

LONG streamType; HWND hPlayWnd; int bNoDecode;

}NET\_SDK\_CLIENTINFO, \*LPNET\_SDK\_CLIENTINFO;

**Members**

*lChannel*

channel number,start from 0 *streamType*

data stream type,two types:NET\_SDK\_MAIN\_STREAM and NET\_SDK\_SUB\_STREAM

*hPlayWnd*

play window handle *bNoDecode*

0:decode,1:not decode.only for windows os,default value is 0

Client SDK Instructions

**NET\_SDK\_DEVICE\_DISCOVERY\_INFO**

discover device automatically on LAN

struct \_net\_sdk\_device\_discovery\_info{ unsigned long deviceType;

char productType[16]; char strIP[16];

char strNetMask[16]; char strGateWay[16]; unsigned char byMac[8];

unsigned short netPort; unsigned short httpPort; unsigned long softVer; unsigned long softBuildDate; }NET\_SDK\_DEVICE\_DISCOVERY\_INFO;

**Members**

*deviceType*

device type,refer to the follow:

**Type**

NET\_SDK\_DVR NET\_SDK\_DVS NET\_SDK\_IPCAMERA NET\_SDK\_SUPERDVR

NET\_SDK\_DECODER

**Description**

Digital video record Network Video Server IP camera

board card

decoder

*productType[16]* product type

*strIP[16]* IP

*strNetMask[16]* subnet mask

*strGateWay[16]*

gateway *byMac[8]*

MAC address *netPort*

network port *httpPort*

http port *softVer*

software version *softBuildDate*

software build date

Client SDK Instructions

**NET\_SDK\_DEVICEINFO**

structure of device login

struct \_net\_sdk\_deviceinfo{ unsigned char localVideoInputNum; unsigned char audioInputNum; unsigned char sensorInputNum; unsigned char sensorOutputNum;

unsigned long displayResolutionMask; unsigned char videoOuputNum;

unsigned char netVideoOutputNum; unsigned char netVideoInputNum; unsigned char IVSNum;

unsigned char presetNumOneCH; unsigned char cruiseNumOneCH; unsigned char presetNumOneCruise; unsigned char trackNumOneCH; unsigned char userNum;

unsigned char netClientNum; unsigned char netFirstStreamNum; unsigned char deviceType; unsigned char doblueStream; unsigned char audioStream; unsigned char talkAudio; unsigned char bPasswordCheck; unsigned char defBrightness; unsigned char defContrast; unsigned char defSaturation; unsigned char defHue;

unsigned short videoInputNum; unsigned short deviceID; unsigned long videoFormat; unsigned long function[8]; unsigned long deviceIP; unsigned char deviceMAC[8]; unsigned long buildDate; unsigned long buildTime;

char deviceName[36];

char firmwareVersion[36]; char kernelVersion[64]; char hardwareVersion[36]; char MCUVersion[36];

char firmwareVersionEx[100]; //firmware version extension for new product

char deviceProduct[28]; //Device model

}NET\_SDK\_DEVICEINFO, \*LPNET\_SDK\_DEVICEINFO;

**Members**

*localVideoInputNum*

number of local video input channel *audioInputNum*

number of audio input channel *sensorInputNum*

number of sensor input channel *sensorOutputNum*

number of relay output *displayResolutionMask*

monitor optional resolution *videoOuputNum*

number of video output(supportable maximum playback channel number)

*netVideoOutputNum*

maximum channel number of network playback *netVideoInputNum*

channel number of digital single input *IVSNum*

number of smart analytics channel *presetNumOneCH*

number of preset point in each channel *cruiseNumOneCH*

number of cruise line in each channel *presetNumOneCruise*

number of preset point in each cruise line *trackNumOneCH*

track number in each channel *userNum*

user number *netClientNum*

maximum client number *netFirstStreamNum*

channel maximum number of main code stream transmission,namely how many clients check main code stream meanwhile

*deviceType* device type

*doblueStream*

whether provide dual stream *audioStream*

whether provide audio stream *talkAudio*

whether enable talkback function:1 means yes;0 means no

*bPasswordCheck*

whether need to input password *defBrightness*

default brightness *defContrast*

default contrast *defSaturation*

default saturation *defHue*

default hue *videoInputNum*

channel number of video input(add network channel number if local video input)

*deviceID* device ID

*videoFormat*

video format,refer to the following list:

**Type** **Value** DD\_VIDEO\_FORMAT\_NTSC 0x01

DD\_VIDEO\_FORMAT\_PAL 0x02

*function[8]*

function description *deviceIP*

device network address *deviceMAC[8]*

device MAC *buildDate*

building date:year<<16 + month<<8 + mday *buildTime*

building time:hour<<16 + min<<8 + sec *deviceName[36]*

device name *firmwareVersion[36]*

firmware version *kernelVersion[64]*

kernel version *hardwareVersion[36]*

hardware version *MCUVersion[36]*

MCU version firmwareVersionEx[100]

Reserved characters deviceProduct[28] Device model

Client SDK Instructions

**NET\_SDK\_EVENT**

struct of event log

struct \_net\_sdk\_event{ unsigned short chnn; unsigned short type; DD\_TIME startTime; DD\_TIME endTime;

}NET\_SDK\_EVENT,\*LPNET\_SDK\_EVENT;

**Members**

*chnn*

event happend in which channel *type*

event type *startTime*

event starting time *endTime*

event ending time

Client SDK Instructions

**NET\_SDK\_FRAME\_INFO**

struct of data frame information

struct \_net\_sdk\_frame\_info{ unsigned long deviceID; unsigned long channel; unsigned long frameType; unsigned long length; unsigned long keyFrame; unsigned long width; unsigned long height; unsigned long frameIndex;

unsigned long frameAttrib; unsigned long streamID; LONGLONG time; LONGLONG relativeTime; }NET\_SDK\_FRAME\_INFO;

**Members**

*deviceID* device ID

*channel*

data channel,channel number starts from 0 *frameType*

data frame type,refer to DD\_FRAME\_TYPE:

**Type** **Value**

DD\_FRAME\_TYPE\_NONE 0x00 DD\_FRAME\_TYPE\_VIDEO 0x01 DD\_FRAME\_TYPE\_AUDIO 0x02 DD\_FRAME\_TYPE\_TALK\_AUDIO 0x03 DD\_FRAME\_TYPE\_JPEG 0x04 DD\_FRAME\_TYPE\_VIDEO\_FORMAT 0x05 DD\_FRAME\_TYPE\_AUDIO\_FORMAT 0x06 DD\_FRAME\_TYPE\_TALK\_AUDIO\_FORMAT0x07

DD\_FRAME\_TYPE\_EVENT 0x08 DD\_FRAME\_TYPE\_TEXT 0x09 DD\_FRAME\_TYPE\_END

*length*

data length *keyFrame*

keyframe,0:minor frame;1:key frame *width*

width of data frame *height*

height of data frame *frameIndex*

data frame index *frameAttrib*

data frame attribute,refer to DD\_FRAME\_ATTRIB:

**Type** **ValueDescription**

DD\_PLAY\_FRAME\_NO\_SHOW 0x01

DD\_PLAY\_FRAME\_SHOW 0x02

DD\_PLAY\_FRAME\_ALL\_END 0x04

DD\_PLAY\_FRAME\_SEC\_END 0x08

DD\_PLAY\_FRAME\_NO\_TIME\_STAMP 0x10

DD\_PLAY\_FRAME\_FF 0x20

no show the frame

show the frame

data read is finished,no more data

the section ends

the frame with time stamp, so shield time function when capture

fast forward frame

live main DD\_LIVE\_FRAME\_FIRST\_STREAM 0x40 code stream

frame

live sub DD\_LIVE\_FRAME\_SECOND\_STREAM0x80 code stream

frame

DD\_LIVE\_FRAME\_JPEG

DD\_LIVE\_FRAME\_TALK

0x100JPEG image

talkback 0x200audio data

frame

*streamID*

data stream ID *time*

absolute time,calculate from 00:00:00 on Jan.1st in 1970,in microsecond,it changes when change device time

*relativeTime*

relative time,in microsecond,it won't change when change device time,because it is continuous

Client SDK Instructions

**NET\_SDK\_JPEGPARA**

struct of JPEG image

struct{

WORD wPicSize; WORD wPicQuality;

}NET\_DVR\_JPEGPARA,\*LPNET\_DVR\_JPEGPARA;

**Members**

*wPicSize*

Picture size：0-CIF，1-QCIF，2-D1，3-UXGA，4-SVGA，5-HD720p，6-VGA，7-XVGA，8-HD900p

*wPicQuality*

Picture quality level：0-best，1-better，2-ordinary

Client SDK Instructions

**NET\_SDK\_LOG**

struct of device log information.

struct{

[DD\_TIME](#_page_635_0) strLogTime; DWORD dwMajorType; DWORD dwMinorType;

char sNetUser[MAX\_NAMELEN]; DWORD dwRemoteHostAddr;

char sContent[MAX\_CONTENTLEN]; }NET\_SDK\_LOG,\*LPNET\_SDK\_LOG;

**Members**

*strLogTime* time of log.

*dwMajorType* main type.

*dwMinorType* minor type.

*sNetUser[MAX\_NAMELEN]* network user.

*dwRemoteHostAddr* remote host address.

*sContent[MAX\_CONTENTLEN]* Details.

Client SDK Instructions

**NET\_SDK\_REC\_EVENT**

struct of record file information according to event.

struct \_net\_sdk\_rec\_event{

DWORD dwChannel; DD\_TIME startTime; DD\_TIME stopTime; DWORD dwRecType; }NET\_SDK\_REC\_EVENT;

**Members**

*dwChannel* channel number.

*startTime* start time.

*stopTime* stop time.

*dwRecType*

record event type,refer to DD\_RECORD\_TYPE:

**Type** **Value Description**

DD\_RECORD\_TYPE\_NONE

DD\_RECORD\_TYPE\_MANUAL

0x0000no record

0x0001manually

DD\_RECORD\_TYPE\_SCHEDULE0x0002regular time

DD\_RECORD\_TYPE\_MOTION

DD\_RECORD\_TYPE\_SENSOR

motion 0x0004detection

record

sensor 0x0008alarm

record

behaviour DD\_RECORD\_TYPE\_BEHAVIOR 0x0010analysis

record

Client SDK Instructions

**NET\_SDK\_REC\_FILE**

struct of record file information.

struct \_net\_sdk\_rec\_file{

DWORD dwChannel; DWORD bFileLocked; DD\_TIME startTime; DD\_TIME stopTime; DWORD dwRecType; DWORD dwPartition; DWORD dwFileIndex; }NET\_SDK\_REC\_FILE;

**Members**

*dwChannel* channel number.

*bFileLocked*

whether record file is locked. *startTime*

start time. *stopTime*

stop times. *dwRecType*

record event type,refer to DD\_RECORD\_TYPE:

**Type** **Value Description**

DD\_RECORD\_TYPE\_NONE

DD\_RECORD\_TYPE\_MANUAL

0x0000no record

0x0001manually

DD\_RECORD\_TYPE\_SCHEDULE0x0002regular time

DD\_RECORD\_TYPE\_MOTION

DD\_RECORD\_TYPE\_SENSOR

motion 0x0004detection

record

sensor 0x0008alarm

record

behaviour DD\_RECORD\_TYPE\_BEHAVIOR 0x0010analysis

record

*dwPartition*

record file partition. *dwFileIndex*

index of record filename.

Client SDK Instructions

**NET\_SDK\_REC\_TIME**

struct of record file information by time.

struct \_net\_sdk\_rec\_time{ DWORD

DD\_TIME DD\_TIME

}NET\_SDK\_REC\_TIME;

dwChannel; startTime; stopTime;

**Members**

*dwChannel* channel number.

*startTime*

the start time of videotape. *stopTime*

the stop time of videotape.

Client SDK Instructions

**NET\_SDK\_RECORD\_STATUS**

struct of record status

struct \_net\_sdk\_record\_status{ DWORD dwRecordType; DWORD dwChannel; }NET\_SDK\_RECORD\_STATUS;

**Members**

*dwRecordType*

record event type,refer to DD\_RECORD\_TYPE:

**Type** **Value Description**

DD\_RECORD\_TYPE\_NONE

DD\_RECORD\_TYPE\_MANUAL

0x0000no record

0x0001recordl

DD\_RECORD\_TYPE\_SCHEDULE0x0002regular

DD\_RECORD\_TYPE\_MOTION

DD\_RECORD\_TYPE\_SENSOR

motion 0x0004detection

record

sensor 0x0008alarm

record

behaviour DD\_RECORD\_TYPE\_BEHAVIOR 0x0010analysis

record

*dwChannel* record channel

Client SDK Instructions

**NET\_SDK\_RECORD\_STATUS\_EX**

struct of record status

struct \_net\_sdk\_record\_status\_ex{ DWORD dwRecordType; DWORD dwChannel;

DWORD dwRecordStatus; }NET\_SDK\_RECORD\_STATUS\_EX;

**Members**

*dwRecordType*

record event type,refer to DD\_RECORD\_TYPE:

**Type** **Value Description**

DD\_RECORD\_TYPE\_NONE

DD\_RECORD\_TYPE\_MANUAL

0x0000no record

0x0001recordl

DD\_RECORD\_TYPE\_SCHEDULE0x0002regular

DD\_RECORD\_TYPE\_MOTION

DD\_RECORD\_TYPE\_SENSOR

motion 0x0004detection

record

sensor 0x0008alarm

record

behaviour DD\_RECORD\_TYPE\_BEHAVIOR 0x0010analysis

record

*dwChannel*

record channel *dwRecordStatus*

0 stoped,1 recording,2 abnormal

Client SDK Instructions

**PTZ\_3D\_POINT\_INFO**

information about PTZ 3D control

struct PTZ\_3D\_POINT\_INFO{ int selBeginX;

int selBeginY; int selEndX; int selEndY;

int displayWidth; int displayHeight; int reserve[2]; }PTZ\_3D\_POINT\_INFO;

**Members**

*selBeginX*

X coordinates of the starting point *selBeginY*

Y coordinates of the starting point *selEndX*

End point X coordinates *selEndY*

End point Y coordinates *displayWidth*

Image width *displayHeight*

Image height *reserve[2]*

Retention value,not enable

**Remarks**

4 coordinate variable is the mouse position relative to the current window of the upper left corner of the screen. enlarge:selBeginX < selEndX, narrow:selBeginX > selEndX

Client SDK Instructions

**NET\_SDK\_IPC\_DEVICE\_INFO**

struct of IPC in device management

\_net\_sdk\_ipc\_device\_info\_{ unsigned long deviceID; unsigned short channel; unsigned short status;

char szEtherName[16]; char szServer[64]; unsigned short nPort;

unsigned short nHttpPort; unsigned short nCtrlPort; char szID[64]; char username[36];

unsigned long manufacturerId;

char manufacturerName[36]; char productModel[36]; unsigned char bUseDefaultCfg; unsigned char bPOEDevice;

unsigned char resv[2]; }NET\_SDK\_IPC\_DEVICE\_INFO;

**Members**

*deviceID*

device ID(reserved) *channel*

Channel number of IPC(Start from 0) *status*

Connection status(1 means online,0 means offline) *szEtherName[16]*

If it is null,default is eth0 *szServer[64]*

IP address of IPC *nPort*

Port of IPC *nHttpPort*

http port

*nCtrlPort*

Control ports, generally the same as the nPort *szID[64]*

Device identification (or MAC address) *username[36]*

username *manufacturerId*

(reserved) *manufacturerName[36]*

(reserved) *productModel[36]*

(reserved) *bUseDefaultCfg*

(reserved) *bPOEDevice*

(reserved) *resv[2]*

(reserved)

Client SDK Instructions

**NET\_SDK\_SEARCH\_IMAGE\_ITEM**

searched face picture information

typedef struct \_net\_sdk\_search\_image\_item\_ {

DD\_TIME\_EX recStartTime; DD\_TIME\_EX recEndTime;

unsigned int similarity; //similarity

unsigned int faceFeatureId; //the matched feature when searching by face features

NET\_SDK\_FACE\_IMG\_INFO\_CH sfaceImg; //the matched picture when searching by face image

unsigned char resv[4];//reserved }NET\_SDK\_SEARCH\_IMAGE\_ITEM;

**Members**

*recStartTime*

start time of face based recording *recEndTime*

end time of face based recording *similarity*

similarity *faceFeatureId*

target face ID *sfaceImg*

matched image information *resv*

reserved

Client SDK Instructions

**NET\_SDK\_SEARCH\_IMAGE\_BY\_IMAG E\_LIST**

the return information of search image by image

typedef struct \_net\_sdk\_search\_image\_by\_image\_list\_ {

unsigned int bEnd; //1 indicates finishing searching images; 0 means there are still images behind.

unsigned int listNum;//return NET\_SDK\_SEARCH\_IMAGE\_ITEM num

NET\_SDK\_SEARCH\_IMAGE\_ITEM \*pSearchImageItem; }NET\_SDK\_SEARCH\_IMAGE\_BY\_IMAGE\_LIST;

**Members**

*bEnd*

Whether all return or not? *listNum*

the number of return data *pSearchImageItem*

return the searched face information

Client SDK Instructions

**NET\_SDK\_CH\_SNAP\_FACE\_IMG\_LIS T**

captured face picture data of a camera

typedef struct \_net\_sdk\_ch\_snap\_face\_img\_list\_ {

unsigned int bEnd; //1 indicates finishing searching images; 0 means there are still images behind.

unsigned int listNum;//return NET\_SDK\_FACE\_IMG\_INFO\_CH num

NET\_SDK\_FACE\_IMG\_INFO\_CH \*pCHFaceImgItem; }NET\_SDK\_CH\_SNAP\_FACE\_IMG\_LIST;

**Members**

*bEnd*

Whether all return or not? *listNum*

the number of return data *pSearchImageItem*

return the searched face information

Client SDK Instructions

**DD\_TIME\_EX**

Struct of time configuration information of the device. Compared to DD\_TIME，the month and year value of DD\_TIME is different .

typedef struct \_dd\_time\_ex\_ {

unsigned char second; minute (0–59)

unsigned char minute; hour (0–59)

unsigned char hour; midnight (0–23)

unsigned char wday; Sunday = 0)

unsigned char mday; 31)

unsigned char month; January = 1)

unsigned short year; year )

int //total seconds

//Seconds after

//Minutes after

//Hours since

//Day of week (0–6;

//Day of month (1–

//Month (1–12;

//Year (current

nTotalseconds;

int //microsecond

}DD\_TIME\_EX;//Compared to of DD\_TIME is different.

nMicrosecond;

DD\_TIME，the month and year value

**Members**

*second*

second; it ranges from 0 to 59. *minute*

minute; it ranges from 0 to 59. *hour*

Count from 0 0'clock. It ranges from 0 to 23. *wday*

Day (it ranges from 0 to 6)，A week starts from sunday (the corresponding value is 0).

*mday*

date of a month. It ranges from 1 to 31. *month*

month. It ranges from 1 to 12. It starts from January. The coresponding value of January is 1.

*year*

Year, the current year

Client SDK Instructions

**DD\_NETWORK\_PLATFORM**

Reigster the upper-level platform

typedef struct \_network\_platform {

//N9000 supports two platforms: national standard platform and platform software

unsigned int CurrentPlat; //the current platform. default: 1 (it indicates platform software)，2 (it indicates national standard platform)

//platform software

unsigned int Switcher; //1 indicates "Enable"， 0 indicates "Disable"

unsigned int Port; //port unsigned int ReportId; //device ID

char szAddress[16]; //ip address //national standard platform，ipc unavailable，

N9000 available

unsigned int SwitchGB; "Enable"，0 indicates "Disable"

unsigned int PortGB;

unsigned int uLocalPort; char szRelm[16];

domain

char szAddressGB[16];

char szUserName[16];

char szPassword[16];

//11 indicates

//port

//local port

//sip server

// address

//username

//password

char szDeviceIdGB[32]; char szServerIdGB[32];

}DD\_NETWORK\_PLATFORM;

//device ID //sip server ID

**Members**

*CurrentPlat*

the current platform. default: 1 (it indicates platform software)，2 (it indicates national standard platform)

Switcher

platform software; 1 indicates "Enable"，0 indicates "Disable"

Port

platform software; port *ReportId*

platform software; device ID szAddress

platform software; ip address *SwitchGB*

national standard platform; 1 indicates "Enable"，0 indicates "Disable

PortGB

national standard platform; port *uLocalPort*

national standard platform; local port

*szRelm*

national standard platform; sip server domain *szAddressGB*

national standard platform; address *szUserName*

national standard platform; username szPassword

national standard platform; password *szDeviceIdGB*

national standard platform; *szServerIdGB*

national standard platform;

device ID

sip server ID

Client SDK Instructions

**DD\_SMART\_VFD\_CONFIG**

Face comparison configuration

typedef struct \_dd\_smart\_vfd\_config\_ {

unsigned int iSize; //struct size

unsigned char enableFaceDetect; //enable/disable face detection

unsigned char enableSaveFacePicture; //enable/disable "Save Face Picture"

unsigned short enableSaveSourcePicture; //enable/disable "Save Source Picture"

unsigned int holdTime; //hold time

DD\_POSITION startPoint; //coordinate information of the upper left point of the rectangle

DD\_POSITION endPoint; //coordinate information of the bottom right point of the rectangle

unsigned int pushModeType; //snapshot mode: 0：auto; it will not capture repeatedly. 1：capture pictures according to the fixed time interval.

unsigned int intervalTime; //inteval period of snapshot (seconds)，only when the snapshot mode is 1, will it take effect. }DD\_SMART\_VFD\_CONFIG;

**Members**

*iSize*

struct size *enableFaceDetect*

enable/disable face detection *enableSaveFacePicture*

enable/disable "Save Face Picture" enableSaveSourcePicture enable/disable "Save Source Picture" holdTime

hold time *startPoint*

coordinate information of the upper left point of the rectangle

*endPoint*

coordinate information of the bottom right point of the rectangle

pushModeType

snapshot mode: 0：auto; it will not capture repeatedly. 1：capture pictures according to the fixed time interval.

*intervalTime*

inteval period of snapshot (seconds)，only when the snapshot mode is 1, will it take effect.

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_GROUP\_ITEM**

struct of the group of face comarison

typedef struct \_net\_sdk\_face\_info\_group\_item\_ {

unsigned char guid[48]; //GROUP GUID

char name[DD\_MAX\_NAME\_LEN];//GROUP NAME

unsigned int property; //NET\_SDK\_FACE\_INFO\_GROUP\_PROPERTY\_TYPE

unsigned int groupId; // unsigned int enableAlarmSwitch;

}NET\_SDK\_FACE\_INFO\_GROUP\_ITEM;

**Members**

*guid*

group GUID *name*

group name *property*

type of the group; refer to NET\_SDK\_FACE\_INFO\_GROUP\_PROPERTY\_TYPE.

*groupId* group id

*enableAlarmSwitch*

Support face match alarm

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_GROUP\_ADD**

Add the group of face comparison

typedef struct \_net\_sdk\_face\_info\_group\_add\_ {

char name[DD\_MAX\_NAME\_LEN];//GROUP NAME

unsigned int property; //NET\_SDK\_FACE\_INFO\_GROUP\_PROPERTY\_TYPE }NET\_SDK\_FACE\_INFO\_GROUP\_ADD;

**Members**

*name*

group name *property*

type of the group; refer to NET\_SDK\_FACE\_INFO\_GROUP\_PROPERTY\_TYPE

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_GROUP\_DEL**

Delete the group of face comparison

typedef struct \_net\_sdk\_face\_info\_group\_del\_ {

unsigned char guid[48]; //GROUP GUID }NET\_SDK\_FACE\_INFO\_GROUP\_DEL;

**Members**

*guid*

Delet the GUID of the group

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_LIST\_GET**

struct of searching face comparison target

typedef struct \_net\_sdk\_face\_info\_list\_get\_ {

unsigned int (compulsory)

unsigned int

pageIndex;

pageSize;

// 1、2、3...

//compulsory

unsigned int groupId;//1、2、3....(compulsory)

char name[DD\_MAX\_NAME\_LEN];//name of NET\_SDK\_FACE\_INFO\_LIST\_ITEM

unsigned int itemId; // itemID of NET\_SDK\_FACE\_INFO\_LIST\_ITEM

}NET\_SDK\_FACE\_INFO\_LIST\_GET;

**Members**

*pageIndex* page number

*pageSize*

the number of items in the page *groupId*

group Id。 *name*

target name (non-compulsory) *itemId*

target ID (non-compulsory)

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_LIST\_ITEM\_G ROUPS**

The group the face picture belongs to and the terms of validity the face picture lasts in this group

typedef struct \_net\_sdk\_face\_info\_list\_item\_groups\_ {

unsigned int groupId; //There is no guid when GROUP id gets target list.

unsigned char guid[48]; //Add GROUP GUID. when target information is edited, guid must be used.

DD\_TIME\_EX validStartTime;//when the property state is "limited", the validStartTime and validEndTime are shown.

DD\_TIME\_EX validEndTime;//when the property state is "limited", the validStartTime and validEndTime are shown. }NET\_SDK\_FACE\_INFO\_LIST\_ITEM\_GROUPS;

**Members**

*groupId* group Id

*guid*

GUID of the group。 *validStartTime*

valid start time of the group; when the type of the group is "limited", it takes effect.

validEndTime

valid end time of the group; when the type of the group is "limited", it takes effect.

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_LIST\_ITEM**

target face information of face comparsion

typedef struct \_net\_sdk\_face\_info\_list\_item\_ {

unsigned int itemId; //id char name[DD\_MAX\_NAME\_LEN];

//compulsory

unsigned int sex; //0:male 1:female unsigned int birthday;//eg:19900707

char nativePlace[DD\_MAX\_NAME\_LEN]; //

unsigned int certificateType; //0:idCard char

certificateNum[DD\_MAX\_CERTIFICATE\_NUM]; //

char mobile[20]; // char number[20]; // unsigned int faceImgCount;

NET\_SDK\_FACE\_INFO\_LIST\_ITEM\_GROUPS groups[DD\_MAX\_FACE\_INFO\_GROUPS];

}NET\_SDK\_FACE\_INFO\_LIST\_ITEM;

**Members**

*itemId :target face Id* name: name

*sex : gender*

*birthday :date of birth* nativePlace: native place

*certificateType : certifcate type; 0:idCard*

*certificateNum : certificate number* mobile: phone number

number: nuber

faceImgCount: number of face pictures

*groups :group information; one face picture can be added to a maximum of 16 groups.*

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_LIST**

search the returned face comparison target lsit.

typedef struct \_net\_sdk\_face\_info\_list\_ {

unsigned int totalNum; // unsigned int listNum;//return

NET\_SDK\_FACE\_INFO\_LIST\_ITEM num [NET\_SDK\_FACE\_INFO\_LIST\_ITEM](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/NET_SDK_FACE_INFO_LIST_ITEM.htm)

\*pFaceInfoListItem; }NET\_SDK\_FACE\_INFO\_LIST;

**Members**

*totalNum*

total number of target *listNum<*

number of target *pFaceInfoListItem*

target list

Client SDK Instructions

**NET\_SDK\_FACE\_IMG\_INFO\_CH**

The face picture captured by the camera can be used as target picture of comparison.

typedef struct \_net\_sdk\_face\_img\_info\_ch\_ {

[DD\_TIME\_EX](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/DD_TIME_EX.htm) frameTime; unsigned int imgId;

unsigned int chl; //return value of 255 means the deleted channel.

unsigned char resv[8];//reserved

}NET\_SDK\_FACE\_IMG\_INFO\_CH;

**Members**

*frameTime<* snapshot time

*imgId* image Id

*chl*

snapshot channel

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_ADD**

Add the face picture you want to compare.

typedef struct \_net\_sdk\_face\_info\_add\_ {

[NET\_SDK\_FACE\_INFO\_LIST\_ITEM](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/NET_SDK_FACE_INFO_LIST_ITEM.htm) sFaceInfoItem; unsigned int imgNum;

[NET\_SDK\_FACE\_IMG\_INFO\_CH](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/NET_SDK_FACE_IMG_INFO_CH.htm) sFaceImgInfo[DD\_MAX\_FACE\_INFO\_IMG];//最大5张

unsigned int unsigned int

==1 Valid

haveImgData;//0、1 imgWidth;//haveImgData

unsigned int imgHeight;//haveImgData ==1 Valid

unsigned int imgLen;//haveImgData ==1 Valid

unsigned char \*imgData;//haveImgData ==1 Valid

}NET\_SDK\_FACE\_INFO\_ADD;

**Members**

*sFaceInfoItem*

Face feature information *imgNum*

the number of images in sFaceImgInfo *sFaceImgInfo*

image information *haveImgData*

Picture data of the external importing picture *imgWidth*

picture width of the external importing picture. haveImgData==1 valid

*imgHeight*

picture height of the external importing picture. haveImgData==1valid

*>imgLen<*

picture size of the external importing picture.haveImgData==1valid

*imgData*

picture data. haveImgData==1valid

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_EDIT**

Edit the face picture you want to compare.

typedef struct \_net\_sdk\_face\_info\_edit\_ {

unsigned int delFaceImgs[DD\_MAX\_FACE\_INFO\_IMG];

[NET\_SDK\_FACE\_INFO\_ADD](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/NET_SDK_FACE_INFO_ADD.htm) sFaceInfoItem;

}NET\_SDK\_FACE\_INFO\_EDIT;

**Members**

*delFaceImgs*

Delete face image *sFaceInfoItem*

modified target face information

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_DEL**

Delete the face picture you want to compare.

typedef struct \_net\_sdk\_face\_info\_del\_ {

unsigned int faceInfoListItemId;//NET\_SDK\_FACE\_INFO\_LIST\_ITEM 中的itemId

unsigned int groupsId[DD\_MAX\_FACE\_INFO\_GROUPS];//NET\_SDK\_F ACE\_INFO\_LIST\_ITEM中的itemId-->groups-->groupId

}NET\_SDK\_FACE\_INFO\_DEL;

**Members**

*faceInfoListItemId*

target face Id。 *groupsId*

group Id。

Client SDK Instructions

**NET\_SDK\_FACE\_MATCH\_ALARM\_TRI GGER**

Alarm linkage information of face match alarm for target groups

typedef struct \_net\_sdk\_face\_match\_alarm\_trigger\_ {

unsigned char guid[48]; //GROUP GUID unsigned int groupId; //group ID unsigned char groupSwitch;//enable

unsigned char alarmOutSwitch;//trigger alarm output

unsigned char alarmOut[16];//trigger a maximum of 16 alarm outputs. The index starts from 1.

unsigned char recSwitch;//recording unsigned int recCH[128];//trigger recording

channels/cameras. The index starts from 1. unsigned char snapSwitch;//snapshot

unsigned int snapCH[128];//trigger snapshot channels/camera. The index starts from 1.

unsigned int popVideo;//pop up window

unsigned char unsigned char unsigned char unsigned char

msgPushSwitch; buzzerSwitch; popMsgSwitch; emailSwitch;

}NET\_SDK\_FACE\_MATCH\_ALARM\_TRIGGER;

**Members**

*guid*

Group GUID

*groupId* Group ID

*groupSwitch*

Whether to enable alarm for the group. *alarmOutSwitch*

trigger alarmOut *alarmOut*

Trigger the channels of alarm out. The index of channels starts from 1.

*recSwitch*

Trigger recording *recCH*

Trigger the recording channels. The index of channels starts from 1.

*snapSwitch* Trigger snapshot

*snapCH*

Trigger snapshot channels. The index of channels starts from 1.

*popVideo*

Trigger pop-up video. 0 menas no video pops up. Other number means the video pops up.

*msgPushSwitch*

Trigger message push *buzzerSwitch*

Trigger buzzer *popMsgSwitch*

Trigger pop-up message *emailSwitch*

Trigger email

Client SDK Instructions

**NET\_SDK\_FACE\_MATCH\_ALARM**

Face match alarm

typedef struct \_net\_sdk\_face\_match\_alarm\_ {

unsigned int similarity;// similarity

unsigned int enableCH[128];//【Enable CH】 starts from 1.

unsigned int faceFeatureGroupsNum;//number of face group

[NET\_SDK\_FACE\_MATCH\_ALARM\_TRIGGER](file:///C:/Users/hj/AppData/Local/Temp/CHM%20Editor/%E8%AE%BE%E5%A4%87%E7%BD%91%E7%BB%9CSDK%E5%BC%80%E5%8F%91%E6%89%8B%E5%86%8C_new_43441.5957271065/interface/struct/NET_SDK_FACE_MATCH_ALARM_TRIGGER.htm) \*pFaceMatchAlarmTrigger;

}NET\_SDK\_FACE\_MATCH\_ALARM;

**Members**

*similarity*

face picture similarity *enableCH*

enable channel. It starts from1. *faceFeatureGroupsNum*

Number of linkage alarms of face target groups *sFaceMatchAlarmTrigger*

linkage alarm information of face target groups

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_IMG\_DATA**

Image data of face picture

typedef struct \_net\_sdk\_face\_info\_img\_data\_ {

unsigned int imgLen;//length of face picture

unsigned char \*imgData;//face picture data }NET\_SDK\_FACE\_INFO\_IMG\_DATA;

**Members**

*imgLen*

the length of face picture *imgData*

face picture data

Client SDK Instructions

**NET\_SDK\_FACE\_INFO\_IMG\_GET**

area struct

typedef struct \_net\_sdk\_face\_info\_img\_get\_ {

unsigned int unsigned int

itemId; //target id

index;//start index 1 of faceImgCount

}NET\_SDK\_FACE\_INFO\_IMG\_GET;

**Members**

*itemId* target Id

*index*

The index of faceImgCount starts from 1.

Client SDK Instructions

**DD\_ENCODE\_CONFIG\_EX**

struct of encoding configuration

struct \_dd\_encode\_config\_{ unsigned long iSize; unsigned short resolution; unsigned short rate; unsigned short encodeType; unsigned short quality; unsigned short minBitrate; unsigned short maxBitrate; unsigned short bitrate; //bitrate

unsigned short encodeFormat; //H264or265 coding DD\_VIDEO\_ENCODE\_FORMAT

char recv[14]; //reserved bytes }DD\_ENCODE\_CONFIG;

**Members**

*iSize*

struct size *resolution*

resolution. Refer to DD\_VIDEO\_SIZE\_N9000： *rate*

Frame rate *encodeType*

encoding type. Refer to the following table.

**Type** **ValueDescription**

DD\_VIDEO\_ENCODE\_MODE\_VBR0x01 VBR DD\_VIDEO\_ENCODE\_MODE\_CBR0x02 BR

*quality*

Refer to the following table.

**Type** **ValueDescription**

DD\_IMAGE\_QUALITY\_LOWEST 0x01

DD\_IMAGE\_QUALITY\_LOWER 0x02

DD\_IMAGE\_QUALITY\_LOW 0x03

DD\_IMAGE\_QUALITY\_MEDIUM 0x04

DD\_IMAGE\_QUALITY\_HEIGHTER 0x05

lowest quality

lower quality

low quality

middle quality

higher quality

DD\_IMAGE\_QUALITY\_HEIGHTEST0x06

highest quality

*minBitrate*

lower limit of bitrate; unit:kbps *maxBitrate*

upper limit of bitrate; unit: kbps *bitrate*

bitrate; unit: kbps *encodeFormat*

Encoding type: H264or265. Refer to DD\_VIDEO\_ENCODE\_FORMAT

Client SDK Instructions

**NET\_SDK\_CH\_SNAP\_FACE\_IMG\_LIS T\_SEARCH**

View face pictures.

typedef struct \_net\_sdk\_ch\_snap\_face\_img\_list\_sreach\_ {

DWORD dwChannel;//camera/channel of snapshot

DD\_TIME\_EX startTime; //time

DD\_TIME\_EX

DWORD

DWORD the page

endTime; //time

pageIndex;//page

pageSize;//the number of items in

unsigned char resv[8]; }NET\_SDK\_CH\_SNAP\_FACE\_IMG\_LIST\_SEARCH;

**Members**

Client SDK Instructions

**NET\_SDK\_SEARCH\_IMAGE\_BY\_IMAG E**

Search image by image

typedef struct \_net\_sdk\_search\_image\_by\_image\_ {

unsigned int pageIndex; //compulsive 1、2、3... unsigned int pageSize; //compulsive

unsigned int similarity; //similarity

unsigned int resultCountLimit; //result limit DD\_TIME\_EX startTime;

DD\_TIME\_EX endTime; unsigned int

searchType;//NET\_SDK\_SEARCH\_IMAGE\_BY\_IMAGE\_TYPE struct

{

unsigned int itemId; //target id

} sfaceFeatures;//SEARCH\_IMAGE\_BY\_FACE\_FEATURES NET\_SDK\_FACE\_IMG\_INFO\_CH

sfaceImgs;//SEARCH\_IMAGE\_BY\_FACE\_IMAGES

struct {

unsigned int groupsId; //GROUP Id }sfaceFeatureGroups

;//SEARCH\_IMAGE\_BY\_FACE\_FEATURE\_GROUPS

struct {

unsigned int unsigned int

isContainRecognized; //0 or 1 isContainNotRecognized; //0 or 1

unsigned int groupsId; //GROUP Id }srecognizedFilter

;//SEARCH\_IMAGE\_BY\_RECONGNIZED\_FILTER struct

{

unsigned int imgWidth;// unsigned int imgHeight;// unsigned int imgLen;// unsigned char \*imgData;//

}sfaceImgData;//SEARCH\_IMAGE\_BY\_FACE\_IMAGE\_DATA

}NET\_SDK\_SEARCH\_IMAGE\_BY\_IMAGE;

**Members**

Client SDK Instructions

**DECODE\_FRAME\_INFO**

Decode YUV frame information

struct decode\_frameInfo {

int nWidth; int nHeight;

unsigned int time; unsigned int dwLen; unsigned char \*pData; }DECODE\_FRAME\_INFO;

**Members**

*nWidth*

frame width *nHeight*

frame height *time*

frame time stamp *dwLen*

decode frame length *pData*

decode frame data

Client SDK Instructions

**NET\_SDK\_USB\_BACKUP\_PROCESS\_E X**

the process and status of the saving record to USB device

typedef struct \_usb\_backup\_process\_ex {

DD\_TIME\_EX startTime; // the start time of the record

DD\_TIME\_EX endTime; // the end time of the record

unsigned int dataSize;//MB the size of the record

unsigned char backupPath[64]; // the usb path of the backup

unsigned char creator[36];// the creator of the backup task

unsigned int progress;//0-100, the process of the backup 0-100

unsigned int backupFileFormat;// unsigned int status;//

unsigned int eventType;//

unsigned char chls[64];// the channel of the backup

unsigned int chlNum;// the actual number of the channels

}NET\_SDK\_USB\_BACKUP\_PROCESS\_EX;

**Members**

*startTime*

the start time of the record *endTime*

the end time of the record *dataSize*

MB the size of the record *backupPath*

the usb path of the backup *creator*

the creator of the backup task *progress*

0-100, the process of the backup 0-100 *backupFileFormat*

0 is avi,1 is private format *status*

0 is backuping, 1 is complete *eventType*

refer to DD\_RECORD\_TYPE *chls*

the channels of the backup *chlNum*

the number of the backup channels

Client SDK Instructions

**NET\_SDK\_IVE\_VEHICE\_ITEM\_INFO**

Vehicle number information

typedef struct {

unsigned int begin\_flag; unsigned int data\_type; unsigned int image\_type; unsigned int plateId; unsigned int plateCharCount; char plate[32];

char plateCharConfid[32]; [NET\_SDK\_IVE\_RECT\_T](#_page_801_0) ptPlateCharRect[32]; unsigned in ptWidth;

unsigned int ptHeight; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptLeftTop; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptRightTop; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptLeftBottom; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptRightBottom; unsigned short plateWidth; unsigned short plateHeight; unsigned int plateConfidence; unsigned int plateIntensity; unsigned char plateColor; unsigned char plateStyle;

unsigned char PlateColorRate; unsigned char vehicleColor; unsigned int plateAngleH; unsigned int plateAngleV; unsigned in jpeg\_len; unsigned int jpeg\_vir\_len; char owner[32];

int listType;

unsigned long long beginTime; unsigned long long endTime; unsigned char iVehicleDirect; unsigned char resrv[11];

unsigned int end\_flag; }NET\_SDK\_IVE\_VEHICE\_ITEM\_INFO;

**Members**

*begin\_flag*

start identification，0x5a5a5a5a *data\_type*

0：JPG,1:YUV *image\_type*

0:source image，1：vehicle number *plateId*

ID，just for identification *plateCharCount*

the number of the characters of the vehicle number *plate*

the vehicle number, utf8 codec *plateCharConfid*

the confidence of the vehicle number *ptPlateCharRect*

the coordinate of the upleft of the vehicle number *ptWidth*

the width of the vehicle number（for drawing rectangle

to following the vehicle number plate） *ptHeight*

the height of the vehicle number *ptLeftTop*

coordinate of the upleft of the plate *ptRightTop*

coordinate of the upright of the plate *ptLeftBottom*

coordinate of the downleft of the plate *ptRightBottom*

coordinate of the downright of the plate *plateWidth*

width of the plate *plateHeight*

height of the plate *plateConfidence*

plate confidence *plateIntensity*

plate intensity *plateColor*

the color of the vehicle number plate // 0-blue 1-black 2-yellow 3-white 4-green 5-red 6-gray 7-purple(KISE)

*plateStyle* plate style

*PlateColorRate*

the similarity of the plate color *vehicleColor*

color of the vehicle *plateAngleH*

horizen angle of the plate *plateAngleV*

vertical angle of the plate *jpeg\_len*

the lenth of the jpeg image data *jpeg\_vir\_len*

the total lenth of the jpeg image data *owner*

the name of the vehicle's owner *listType*

list type,0-comparision failed，1-strange，2-white

list，3-black list， *beginTime*

start time *endTime*

end time *iVehicleDirect*

vehicle's direction, 1 unknown 2 near 3 far *resrv*

preserve *end\_flag*

end identification，0xa5a5a5a5

Client SDK Instructions

**NET\_SDK\_IVE\_VEHICE\_HEAD\_INFO**

Vehicle number detection alarm call back information header

typedef struct {

unsigned int begin\_flag; unsigned int item\_cnt; unsigned int plate\_cnt; long long relativeTime; long long absoluteTime;

unsigned int softwareVersion; unsigned int softwareBuildDate; unsigned int resver[2]; unsigned int end\_flag;

}NET\_SDK\_IVE\_VEHICE\_HEAD\_INFO;

**Members**

*begin\_flag*

start identification，0x5a5a5a5a *item\_cnt*

the number of NET\_SDK\_IVE\_VEHICE\_ITEM\_INFO *plate\_cnt*

the number of vihicle number has detected *relativeTime*

the relative time of the detected happened *absoluteTime*

the absolute time of the detected happened *softwareVersion*

the version of the software, 0xABCDEFGH,AB：Brand

CD：main version EFGH：sub version Brand 1:OMRON version:V5.00

*softwareBuildDate*

the build time of the software,0xYYYYMMDD

*resver* Preserve *end\_flag*

end identification，0xa5a5a5a5

Client SDK Instructions

**NET\_SDK\_DEV\_SUPPORT**

the functions of the IPC

struct {

unsigned int supportThermometry:1; //support mask and temperature detect

unsigned int supportVfd:1; //support face detect

unsigned int supportVfdMatch:1; //support face match

unsigned int supportThermal:1; //thermal

unsigned int supportPassLine:1; //pass line

unsigned int supportresv:27; // unsigned int resv[15]; // }NET\_SDK\_DEV\_SUPPORT;

Client SDK Instructions

**REG\_LOGIN\_INFO**

the information of the auto register device。

typedef struct \_reg\_login\_info {

unsigned int deviceId;//register id

char m\_szUserName[36];//user name char m\_szPasswd[36];//password

}REG\_LOGIN\_INFO;

Client SDK Instructions

**SEARCHED\_DEVICE\_INFO**

searched device's information

struct \_searched\_deviceInfo{

char series[64]; char devName[64]; char deviceType[16];

char szproductModel[16]; char szVersion[32];

char szFactoryName[16]; char szEthName[16];

unsigned short netport; unsigned short nHttpPort;

unsigned int ipaddr; unsigned int gateway; unsigned int netmask; unsigned int dns1; unsigned int dns2;

unsigned short nChannelCount; //NVR's channel count

unsigned int dwSecondIP; unsigned int dwSecondMask;

}SEARCHED\_DEVICE\_INFO;

**Members**

*series[64]* series

*devName[64]* device name

*deviceType[16]* device type

*szproductModel[16]* product model

*szVersion[32]* version

*szFactoryName[16]* factory name

*szEthName[16]*

ethnet name *netport;*

net port *nHttpPort;*

http port *ipaddr*

ip address *gateway*

gate way *netmask*

net mask *dns1*

dns1 *dns2*

dns2 *nChannelCount*

NVR channel count *dwSecondIP*

second ip *dwSecondMask*

second mask

Client SDK Instructions

**NET\_SDK\_IVE\_FACE\_MATCH\_ADD\_F ACE\_REPLY\_T**

return struct of the adding face to IPC

typedef struct \_net\_sdk\_ive\_face\_match\_add\_face\_reply\_t {

unsigned int dwResult;

int iPersonId; // person ID.

char szRes[32]; }NET\_SDK\_IVE\_FACE\_MATCH\_ADD\_FACE\_REPLY\_T;

**Members**

*dwResult;* result。

*iPersonId* person ID。

*szRes* reserve。

Client SDK Instructions

**NET\_SDK\_IVE\_PASSLINECOUNT\_T**

pass line information

typedef struct {

unsigned int enterCarCount; //enter car count

unsigned int enterPersonCount;//enter person count

unsigned int enterBikeCount;//enter bike count

unsigned int leaveCarCount; //leave car count, it'll be 0 if single direction

unsigned int leavePersonCount;//leave person count, it'll be 0 if single direction

unsigned int leaveBikeCount;//leave bike count, it'll be 0 if single direction

unsigned int existCarCount;//exist car count, it'll be 0 if single direction

unsigned int existPersonCount;//exist person count, it'll be 0 if single direction

int

existBikeCount;//exist bike count, it'll be 0 if single direction

unsigned int count; //count [NET\_SDK\_IVE\_PASSLINECOUNT\_INFO\_T](#_page_797_0) passLineInfo[32];

// pass line analyse result information }NET\_SDK\_IVE\_PASSLINECOUNT\_T;

Client SDK Instructions

**NET\_SDK\_IVE\_PASSLINECOUNT\_INF O\_T**

pass line analyse result information

typedef struct {

unsigned int eventId; id

unsigned char status; status,0:none 1:start 2:end 3:procedure

unsigned char reserve[3]; reserve

unsigned int targetId; ID

[NET\_SDK\_IVE\_LINE\_T](#_page_799_0) line; line rule

[NET\_SDK\_IVE\_RECT\_T](#_page_801_0) rect; target rectangle }NET\_SDK\_IVE\_PASSLINECOUNT\_INFO\_T;

// event

// alarm

//

// target

// pass

//

Client SDK Instructions

**NET\_SDK\_IVE\_LINE\_T**

pass line rule

typedef struct {

unsigned int X1; unsigned int Y1; unsigned int X2; unsigned int Y2;

}NET\_SDK\_IVE\_LINE\_T;

// start x coodinate // start y coodinate // end x coodinate // end y coodinate

Client SDK Instructions

**NET\_SDK\_IVE\_RECT\_T**

target rectangle

typedef struct {

unsigned int X1; unsigned int Y1; unsigned int X2; unsigned int Y2;

}NET\_SDK\_IVE\_RECT\_T;

// top left x coodinate // top left y coodinate // right down x coodinate // right down y coodinate

Client SDK Instructions

**NET\_SDK\_IVE\_AVD\_T**

struct of abnormal video detection

typedef struct {

unsigned int count; [NET\_SDK\_IVE\_AVD\_INFO\_T](#_page_805_0) avdInfo[32];

}NET\_SDK\_IVE\_AVD\_T

**Members**

*count* count

*avdInfo*

avd information struct array

Client SDK Instructions

**NET\_SDK\_IVE\_AVD\_INFO\_T**

struct of abnormal video detection detail

typedef struct {

unsigned int eventId; unsigned int status; unsigned int type;

}NET\_SDK\_IVE\_AVD\_INFO\_T

**Members**

*eventId* eventId

*status*

0:none 1:start 2:end 3:procedure *type*

0:none 1:Scene 2:Clarity 3:Color

Client SDK Instructions

**NET\_DVR\_IVE\_VFD\_RESULT\_HEAD\_ T**

struct of video face detection result head

typedef struct {

LONGLONG time; LONGLONG relativeTime;

unsigned int detectDataLen; unsigned int softwareVersion; unsigned int softwareBuildDate; unsigned int faceCnt;

unsigned int faceDataLen[40]; }NET\_DVR\_IVE\_VFD\_RESULT\_HEAD\_T

**Members**

*time*

current time *relativeTime*

relative time *detectDataLen*

detect data length *softwareVersion*

software version 0xABCDEFGH,AB:manufacture,CD:major version EFGH:minor version

*softwareBuildDate* software build date

*faceCnt*

face count, max is 40 *faceDataLen*

face data length

Client SDK Instructions

**NET\_DVR\_IVE\_VFD\_RESULT\_DATA\_I NFO\_T**

struct of video face detection result data information

typedef struct

{ unsigned inttype; unsigned int status; unsigned int width; unsigned int height;

unsigned int dataLen; }NET\_DVR\_IVE\_VFD\_RESULT\_DATA\_INFO\_T

**Members**

*type*

0, JPG; 1, YUV *status*

0, INVALID; 1, VALID; 2, SAVED *width*

width *height*

height *dataLen*

data Length

Client SDK Instructions

**NET\_DVR\_IVE\_VFD\_RESULT\_FACE\_D ATA\_INFO\_T**

face data information

typedef struct {

int faceId;

unsigned int ptWidth; unsigned int ptHeight;

[NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptLeftTop; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptRightTop;

[NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptLeftBottom; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) ptRightBottom; int nPose;

int nConfidence; int age;

int sex;

int dtFrames;

int featureSize; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) stPosFaceImg;

float feature\_score;

short eye\_dist; short blur;

char pose\_est\_score; char detect\_score; char illumination; char faceliveness;

char completeness; char glasses;

char wearmask; char reserved1[1];

float comprehensive\_score; int temperature;

int foreheadX;

int foreheadY;

[NET\_SDK\_IVE\_POINT\_T](#_page_829_0) stHotLeftTop; [NET\_SDK\_IVE\_POINT\_T](#_page_829_0) stHotRightBottom; char cTemperatureMode; char tempUnitsType;

char cTemperatureStatus; char reserved[5];

[NET\_DVR\_IVE\_VFD\_RESULT\_DATA\_INFO\_T](#_page_809_0) stFaceImgData; }NET\_DVR\_IVE\_VFD\_RESULT\_FACE\_DATA\_INFO\_T;

**Members**

*faceId*

face ID Number *ptWidth*

width *ptHeight*

height *ptLeftTop*

Left-Top Face Coordinates *ptRightTop*

Right-Top Face Coordinates *ptLeftBottom*

Left-Bottom Face Coordinates *ptRightBottom*

Right-Bottom Face Coordinates *nPose*

Face Pose *nConfidence*

Confidence Degree *age*

age *sex*

sex *dtFrames*

dtFrames *featureSize*

feature size *stPosFaceImg*

the coodinate of the image left top *feature\_score*

feature score 0-100 *eye\_dist*

distance of the eyes *blur*

blur *pose\_est\_score*

pose est\_score 0-100 *detect\_score*

detect score 0-100 *illumination*

illumination *faceliveness*

faceliveness0~100 *completeness*

completeness 0~100 *glasses*

if wear glasses *wearmask*

if wear mask *reserved1*

reserved1 *comprehensive\_score*

comprehensive score [90,100)best， [80,90)better，

[70,80)good，[60,70)normal，[50,60)medium， [0,50)bad。

*temperature* temperature

*foreheadX*

forehead X coordinate *foreheadY*

forehead Y coordinate

*stHotLeftTop*

hot left top Coordinates *stHotRightBottom*

hot right top Coordinates *cTemperatureMode*

0:normal 1:validate *tempUnitsType*

0:Celsius 1: Fahrenheit *cTemperatureStatus*

0:normal，1:temperature too low，2:temperature too high

*reserved* reserved

*stFaceImgData* face image data

Client SDK Instructions

**NET\_SDK\_IVE\_FACE\_MATCH\_T**

struct of face match

typedef struct {

[DD\_TIME\_EX](#_page_721_0) frameTime; unsigned int dwRealFaceID; unsigned int dwGrpID; unsigned int dwLibFaceID; unsigned int dwSimilar; unsigned char byName[32]; unsigned int Channel; unsigned int imgLen;

}NET\_SDK\_IVE\_FACE\_MATCH\_T

**Members**

*frameTime* frameTime

*dwRealFaceID* snap face id

*dwGrpID* group id

*dwLibFaceID* library face id

*dwSimilar* similarity

*byName* name

*Channel* Channel

*imgLen*

image length

Client SDK Instructions

**NET\_SDK\_AVPSTORE\_FACE\_ABSTRA CT\_INFO**

struct of face abstract

typedef struct {

char szName[32]; unsigned int dwBirth; char szNativePlace[16]; char szNote[16]; unsigned char byPicNum;

unsigned char byTypeCredential; unsigned char bySex;

unsigned char byGroupCount; unsigned char byGroupID[4]; union

{

struct {

unsigned int dwStartTime; unsigned int dwReserve[3]; unsigned int dwEndTime; unsigned char byReserve[11]; unsigned char byContentType;

}PeriodV1;

struct {

unsigned unsigned unsigned unsigned unsigned unsigned unsigned

}PeriodV2;

int byWeekOrDate; int dwReserve[3]; short wStartTime; short wEndTime; short wReserve[5]; char byMode;

char byContentType;

struct {

unsigned char unsigned char

}PlaceHolder;

dwReserve[31]; byContentType;

}TimeCycle;

char szCredential[32];

unsigned char byPhoneNum[16]; unsigned char byIDParam[16];

}NET\_SDK\_AVPSTORE\_FACE\_ABSTRACT\_INFO

**Members**

*szName* name

*dwBirth*

birthday like 19991220 *szNativePlace*

native place *szNote*

note *byPicNum*

number of picture *byTypeCredential*

credential type *bySex*

0:male 1:female *byGroupCount*

group count *byGroupID*

group id *ress*

reserve *szCredential*

credential id *byPhoneNum*

phone number *byIDParam*

id

Client SDK Instructions

**NET\_SDK\_TLV\_BUFFER\_DESC**

struct of buffer description

public struct NET\_SDK\_TLV\_BUFFER\_DESC {

unsigned char ucID; unsigned char ucVersion; unsigned short usNumber; unsigned int dwSize;

}

**Members**

*ucID* id

*ucVersion* version

*usNumber* number

*dwSize*

the source image's size

Client SDK Instructions

**NET\_SDK\_IVE\_BASE\_INFO**

struct of IPC face match base information

typedef {

struct NET\_SDK\_IVE\_BASE\_INFO\_T

long long i64SnapTime; unsigned int iSnapPicId;

int iSimilarity; int iPersonId; int iType;

char szName[128]; int iMale;

int iAge;

char szIdentifyNum[128]; char szTel[64];

char szRes[128];

int iSnapPicQuality; int iSnapPicAge;

int iSnapPicSex;

char livingBody; char comparisonRes; char wearmask;

char tempUnitsType; int temperature;

char keyID[36]; char szReserve[20];

}NET\_SDK\_IVE\_BASE\_INFO

**Members**

*i64SnapTime* snap time

*iSnapPicId*

snap picture id *iSimilarity*

(0-100) similarity *iPersonId*

the person's id *iType*

0:stranger 1:white list 2: black list *szName*

name *iMale*

1:male 0:female. *iAge*

age *szIdentifyNum*

identify number *szTel*

telphone number *szRes*

reserve *iSnapPicQuality*

snap picture's quality *iSnapPicAge*

snap picture's age *iSnapPicSex*

snap picture's sex *livingBody*

1:living body 0:not *comparisonRes*

comparision result 1:success 0:failed *wearmask*

if ware mask 0:not detect 1:not wear 2 wear mask *tempUnitsType*

temperature unit type 0:celsius 1:Fahrenheit *temperature*

temperature *keyID*

keyID *szReserve*

reserve

Client SDK Instructions

**NET\_SDK\_IVE\_PICTURE\_INFO**

struct of IPC snap picture information

typedef struct NET\_SDK\_IVE\_PICTURE\_INFO\_T {

int iWidth; int iHeight;

int iPicFormat; int iPicSize;

}NET\_SDK\_IVE\_PICTURE\_INFO

**Members**

*iWidth*

picture's width *iHeight*

picture's height *iPicFormat*

picture's format *iPicSize*

picture's size

Client SDK Instructions

**NET\_SDK\_IVE\_POINT\_T**

point

typedef struct {

int X; int Y;

}NET\_SDK\_IVE\_POINT\_T

**Members**

*X1*

x coodinate *Y1*

y coodinate

Client SDK Instructions

**NET\_SDK\_NVR\_DISKREC\_DATE\_ITE M**

the structure of NVR record days information

typedef struct \_net\_sdk\_nvr\_diskrec\_date\_item {

unsigned int diskCount; unsigned int diskIndex;

char szDiskSizeGB[16]; char szStartDate[32]; char szEndDate[32];

}NET\_SDK\_NVR\_DISKREC\_DATE\_ITEM;

**Members**

*diskCount* disk count.

*diskIndex* disk index.

*szDiskSizeGB*

the size(GB) of the disk. *szStartDate*

recording start day. *szEndDate*

recording end day.

Client SDK Instructions

**NET\_SDK\_DiscoverDevice**

discover device automatically on LAN

long NET\_SDK\_DiscoverDevice( [NET\_SDK\_DEVICE\_DISCOVERY\_INFO](clbr://internal.invalid/book/struct/NET_SDK_DEVICE_DISCOVERY_INFO.html)

*\*pDeviceInfo*, long *bufNum*;

long *waitSeconds*; );

**Parameters**

*\*pDeviceInfo*

[in] an array witch is needed to asign values,its size is **bufNum** ,if descovered device num is more than,the returned size is just **bufNUm**

*bufNum*

[in] size of the array *waitSeconds*

[in] time to discover devices, unit is second,this interface will be returned after **waitSeconds**

**Return Values**

Returned value is the num of discovered devices,if no deivce is found or discovering device gets error,the value is 0. Get error info refer to [NET\_SDK\_GetLastError](clbr://internal.invalid/book/NET_SDK_GetlastError.htm)

**See Also**

[NET\_SDK\_GetDeviceInfo](#_page_467_0)

Client SDK Instructions

**FAQ**

**Q1 How to get alarm means & invoking method?**

**A1**

What is called protection is SDK connects device actively,device starts loading alarm,once alarm happens alarm information is uploaded to SDK as soon as possible.So except that alarm inputs information & device invokes callback function,the interface NET\_SDK\_SetupAlarmChan also should be invoked to setup connection between SDK and device.

**Q2 Alarm configuration has succeed and alarm signal can be received locally,but why can't client receive alarm signal?**

**A2**

Reasons as follows:1)whether network connection is normal 2)when alarm type is protection,whether setup protection correctly.

**Q3 Why is it that returned value is failure when calling NET\_SDK\_SetDVRConfig()?**

**A3**

NET\_SDK\_EnterDVRConfig() must be called to lock config before calling NET\_SDK\_SetDVRConfig().

**Q4 Why is it that the start time of playback and downloading is different from the setting start time?**

**A4**

Playback & download start from the nearby key frame of the setting start time.

**Q5 Why need to pass a group of channel numbers to NET\_SDK\_PlayBackByTime()?**

**A5**

The passed group channels realize autosynchronous play, at the same time divide channels into groups to play but not play by itself, these can save device-side performance.

**Q6 Why the time of record data index,playback and download is different from the time of device-side?**

**A6**

If this problem appears,first check that whether device timezone and PC timezone is the same,and then check that the time of the two machines is the same.

**Q7 What to notice when using configurated parameters in NET\_SDK\_SetDVRConfig()?**

**A7**

Because NET\_SDK\_SetDVRConfig() needs struct with complete assignment,otherwise setting error comes out. So for fear of this error, popularly invoke NET\_SDK\_GetDVRConfig() to assign initial values to the struct which is needed modification before invoking NET\_SDK\_SetDVRConfig().

**Q8 Why does control command of NET\_SDK\_PTZControl have no effect on some certain PTZ?**

**A8**

Device sends control code to PTZ according to decoder type and decoder address.If current decoder unmatchable,matching decoder setup is needed; if device doesnot support the decoder, control command from device has no effect on the PTZ.

**Q9 Do audio talkback & forward aim at device or channel?**

**A9**

Aim at device ,not channel.

**Q10 Whether the callback function of audio talkback can set be null or not.**

**A10**

Yes,if be null,vioce is still normal but user can't access data.

**Q11 How to save record data into files?**

**A11**

Get data through callback function LIVE\_DATA\_CALLBACK of NET\_SDK\_SetLiveDataCallBack,and then save the data into files,refer to the example in livedlg.cpp of SDKdemo,see L1170.You can play the saved files by Player.

**Q12 How to get play progress?**

**A12**

Get start time and end time by NET\_SDK\_PlayBackByTime,and then get current playing time by NET\_SDK\_GetPlayBackOsdTime.Play progress=current playing time/(end time-start time).

**Q13 How to do when PlayerDemo gets error code 0XC0150002?**

**A13**

Solution:install Microsoft Visual C++ 2005 SP1 Redistributable Package4,download the module in MSDN.

**Q14 Why some function are invalid when palyback?**

**A14**

2X and 4X speed is invalid in SDKDemo and SDK when backward,but other speed is OK.When forward all speed is OK except 1X.Before starting playback one frame by one frame,*Pause* should be enabled,and then one frame can be played by click *Next frame* button one time.

**Q15 What is *wday* in DD\_TIME struct?**

**A15**

Start time is DD\_TIME type in NET\_SDK\_FindFile,but *wday* is invalid in DD\_TIME.*wday* can be empty but can not be cleared.You can search by *mday* or write a function to convert time into wday.

**Q16 How to do when play file gets E\_PLAYER\_BAD\_FORMAT\_FILE error?**

**A16**

Check the following four qustions:

1the first frame is format frame when save record file,

2all structs in SDK are 4 bytes alignment,

3check interface calling order,

4the file in PlayerSDK should be :

\*frame info£¨SDK\_FRAME\_INFO£©valid data in frame

\*frame info£¨SDK\_FRAME\_INFO£©valid data in frame

\* ......

\*frame info£¨SDK\_FRAME\_INFO£©valid data in frame

\* ......

\* video info frame should be before video frame,audio info frame should be before audio frame too

Client SDK Instructions

**NET\_SDK\_GetLastError**

return the last error code of operation

DWORD NET\_SDK\_GetLastError( );

**Return Values**

return value is pointer to error code information. error message has two main types,error message of network communication library and error message of soft and hard decoding library,list the first type as follows:

**error message of network communication library**

**type of errors** NET\_SDK\_SUCCESS

**error value**

0 no error

NET\_SDK\_PASSWORD\_ERROR NET\_SDK\_NOENOUGH\_AUTH NET\_SDK\_NOINIT NET\_SDK\_CHANNEL\_ERROR NET\_SDK\_OVER\_MAXLINK NET\_SDK\_LOGIN\_REFUSED NET\_SDK\_VERSION\_NOMATCH NET\_SDK\_NETWORK\_FAIL\_CONNECT NET\_SDK\_NETWORK\_NOT\_CONNECT NET\_SDK\_NETWORK\_SEND\_ERROR NET\_SDK\_NETWORK\_RECV\_ERROR NET\_SDK\_NETWORK\_RECV\_TIMEOUT NET\_SDK\_NETWORK\_ERRORDATA NET\_SDK\_ORDER\_ERROR NET\_SDK\_OPER\_BY\_OTHER NET\_SDK\_OPER\_NOPERMIT NET\_SDK\_COMMAND\_TIMEOUT NET\_SDK\_ERROR\_SERIALPORT NET\_SDK\_ERROR\_ALARMPORT NET\_SDK\_PARAMETER\_ERROR NET\_SDK\_CHAN\_EXCEPTION NET\_SDK\_NODISK NET\_SDK\_ERROR\_DISKNUM NET\_SDK\_DISK\_FULL NET\_SDK\_DISK\_ERROR NET\_SDK\_NOSUPPORT NET\_SDK\_BUSY NET\_SDK\_MODIFY\_FAIL

NET\_SDK\_PASSWORD\_FORMAT\_ERROR NET\_SDK\_DISK\_FORMATING NET\_SDK\_DVR\_NORESOURCE NET\_SDK\_DVR\_OPRATE\_FAILED NET\_SDK\_OPEN\_HOSTSOUND\_FAIL NET\_SDK\_DVR\_VOICEOPENED NET\_SDK\_TIME\_INPUTERROR NET\_SDK\_NOSPECFILE NET\_SDK\_CREATEFILE\_ERROR NET\_SDK\_FILEOPENFAIL NET\_SDK\_OPERNOTFINISH NET\_SDK\_GETPLAYTIMEFAIL NET\_SDK\_PLAYFAIL

1 user's name or pa 2 no right for this o 3 SDK is not initial 4 error of channel n 5 the client connect 6 SDK login is refu 7 version doesn't m 8 failed to connect t 9 network isn't conn 10 failed to send data 11 failed to receive t 12 timeout when rec 13 send illegal data t 14 the called order er 15 operation method 16 the privileged use 17 DVR command ti 18 error of serial por 19 error of alarm por 20 parameter error 21 server's channel is 22 no hard disk

23 hard disk no. erro 24 server hark disk is 25 server hard disk e 26 server does not su 27 server is busy

28 failed to modify i 29 the password inpu 30 hard disk is forma 31 DVR no resource 32 DVR failed to ope 33 failed open PC vo 34 server voice dialo 35 time input is not c 36 there is no appoin 37 failed to create a f 38 faile to open a file 39 the last operation 40 faile to get the cur 41 failed to play

NET\_SDK\_FILEFORMAT\_ERROR NET\_SDK\_DIR\_ERROR NET\_SDK\_ALLOC\_RESOURCE\_ERROR NET\_SDK\_AUDIO\_MODE\_ERROR NET\_SDK\_NOENOUGH\_BUF NET\_SDK\_CREATESOCKET\_ERROR NET\_SDK\_SETSOCKET\_ERROR NET\_SDK\_MAX\_NUM NET\_SDK\_USERNOTEXIST NET\_SDK\_WRITEFLASHERROR NET\_SDK\_UPGRADEFAIL NET\_SDK\_CARDHAVEINIT NET\_SDK\_PLAYERFAILED NET\_SDK\_MAX\_USERNUM

NET\_SDK\_GETLOCALIPANDMACFAIL NET\_SDK\_NOENCODEING NET\_SDK\_IPMISMATCH NET\_SDK\_MACMISMATCH NET\_SDK\_UPGRADELANGMISMATCH NET\_SDK\_MAX\_PLAYERPORT NET\_SDK\_NOSPACEBACKUP NET\_SDK\_NODEVICEBACKUP NET\_SDK\_PICTURE\_BITS\_ERROR NET\_SDK\_PICTURE\_DIMENSION\_ERROR NET\_SDK\_PICTURE\_SIZ\_ERROR NET\_SDK\_LOADPLAYERSDKFAILED NET\_SDK\_LOADPLAYERSDKPROC\_ERROR NET\_SDK\_LOADDSSDKFAILED NET\_SDK\_LOADDSSDKPROC\_ERROR NET\_SDK\_DSSDK\_ERROR NET\_SDK\_VOICEMONOPOLIZE NET\_SDK\_JOINMULTICASTFAILED NET\_SDK\_CREATEDIR\_ERROR NET\_SDK\_BINDSOCKET\_ERROR NET\_SDK\_SOCKETCLOSE\_ERROR NET\_SDK\_USERID\_ISUSING NET\_SDK\_PROGRAM\_EXCEPTION NET\_SDK\_WRITEFILE\_FAILED NET\_SDK\_FORMAT\_READONLY NET\_SDK\_WITHSAMEUSERNAME NET\_SDK\_DEVICETYPE\_ERROR NET\_SDK\_LANGUAGE\_ERROR NET\_SDK\_PARAVERSION\_ERROR NET\_SDK\_FILE\_SUCCESS NET\_SDK\_FILE\_NOFIND

42 the file input form 43 path error

44 resources allotting 45 display card mode 46 buffer is not enou 47 establish SOCKE 48 set SOCKET erro 49 the max number 50 user doest not exi 51 wirte FLASH erro 52 failed to upgrade 53 the decode card is 54 player failed

55 the max user no. failed to get the IP

56 end or physical ad 57 the channel is not 58 IP address not ma 59 MAC address not 60 the language of up 61 reach to the max p 62 no enough space t 63 no backup device 64 the bits of picture 65 the dimension is o 66 the size of picture 67 failed to load play 68 not find some fun 69 failed to load DsS 70 not find some fun 71 failed to call func 72 voice card is mon 73 failed join to mult 74 failed to create lo 75 failed to bind sock 76 socket is closed 77 the user ID is ope 78 sdk program exce 79 write file failed

80 failed to format re 81 there is same user 82 device type no ma 83 language no matc 84 soft version no m 85 file has been crea 86 file isn't found

NET\_SDK\_NOMOREFILE 87 NET\_SDK\_FILE\_EXCEPTION 88 NET\_SDK\_TRY\_LATER 89 NET\_SDK\_DEVICE\_OFFLINE 90 NET\_SDK\_CREATEJPEGSTREAM\_FAIL 91 NET\_SDK\_USER\_ERROR\_NO\_USER 92 NET\_SDK\_USER\_ERROR\_USER\_OR\_PASSWORD\_IS\_NULL 93 NET\_SDK\_USER\_ERROR\_ALREDAY\_LOGIN 94 NET\_SDK\_USER\_ERROR\_SYSTEM\_BUSY 95 NET\_SDK\_DEVICE\_NOT\_SUPPROT 96 NET\_SDK\_USER\_ERROR\_SYSTEM\_NO\_READY 97 NET\_SDK\_CHANNEL\_OFFLINE 98 NET\_SDK\_GETREADYINFO\_FAIL 99 NET\_SDK\_NORESOURCE 100 NET\_SDK\_DEVICE\_QUERYSYSTEMCAPS\_FAIL 101 NET\_SDK\_INBUFFER\_TOSMALL 102 NET\_SDK\_NO\_PASSWORD\_STRENGTH 103

there is no more f file exception Try again later Device offline

Failed to create JP No such user!

No username or p The user has been The device is bus The device don no Do not complete g Camera is offline It fails to get devi SDK resources is The device fails to The input buffer a The password stre

**Remarks**

Get error number through NET\_SDK\_GetErrorMsg

**See Also**

[NET\_SDK\_GetErrorMsg](#_page_131_0)