RISS.Device.Dll User’s Guide

Catalog

Chapter 1 Entity class definition

1.1. Device

namespace：Riss.Devices

Description：Device Entity Class

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	DN	Device ID	Int		
2.	SerialNumber	Device serial number	String		
3.	Model	Device Model	String		
4.	CommunicationType	Communication type	CommunicationType	0-2	0：Serial Port 1：TCP/IP 2：USB
5.	Baudrate	Baud rate of Serial Communication	Int		
6.	SerialPort	Serial Communication port number	Int		
7.	Password	Communication password	String		
8.	IpPort	TCP/IP Communication port number	Int		
9.	IpAddress	TCP/IP Communication IP address	String		
10.	Label	Devices displaying text information	String		
11.	ConnectionModel	Connection type	Int	5	ZD2911 Platform

1.2. User

namespace：Riss.Devices

Description：User Entity Class

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	Privilege	User privilege	Int	1、2、4、8、16	1：User 2：Registrar

					4：LogQuery 8：Manager 16 ： Client (Reserve)
2.	DIN	Device ID	UInt64		Max=18 digits
3.	UserName	User Name	String		
4.	IDNumber	ID Number	String		
5.	Sex	Gender	Sex		
6.	Enable	Enable	Bool		
7.	Comment	Remark Information (“Notes” in the enrollment info.)	String		
8.	DeptId	Department ID	String		
9.	AttType	Att. duty time No. of the device.(Reserve)	Int		
10.	Birthday	Date of Birth	DateTime		
11.	AccessControl	Enable lock 1, lock 2 or not?	Int	0-3	0：Disable 1：Enable lock 1 2：Enable lock 2 3：Enable both lock 1 and lock 2
12.	ValidityPeriod	Enable user validity period or not? (Access control function)	Bool		
13.	UseUserGroupACTZ	Enable user group for access control time zone or not?	Bool		
14.	UseUserGroupVM	Enable the user group verification mode for access control or not?	Bool		
15.	Department	Department NO. of the device	Int		
16.	Enrolls	Collection of user enrollment information	List<Enroll>		
17.	AccessTimeZone	Access time zone for access control	Int		
18.	ValidDate	Valid Date	DateTime	2010-01-01 to 2099-12-3	

				1	
19.	InvalidDate	Invalid Date	DateTime	2010-01-01 to 2099-12-31	
20.	UserGroup	User group of the device	Int		
21.	LockControl	scope of control of the door lock	UInt16		

1.3. UserExt

namespace：Riss.Devices

Description：User Entity Class, for the calls from those don't support the generic

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	Privilege	User privilege	Int	1、2、4、8、16	1：User 2：Registrar 4：LogQuery 8：Manager 16：Client (Reserve)
2.	DIN	Device ID	String		Max=18 digits
3.	UserName	User Name	String		
4.	IDNumber	ID Number	String		
5.	Sex	Gender	Sex		
6.	Enable	Enable	Bool		
7.	Comment	Remark Information (“Notes” in the enrollment info.)	String		
8.	DeptId	Department ID	String		
9.	AttType	Att. duty time No. of the device.(Reserve)	Int		
10.	Birthday	Date of Birth	DateTime		
11.	AccessControl	Enable lock 1, lock 2 or not?	Int	0-3	0：Disable 1：Enable lock 1 2：Enable lock 2 3：Enable both lock 1 and lock 2
12.	ValidityPeriod	Enable user validity period or not?	Bool		

		(Access control function)			
13.	UseUserGroupACTZ	Enable user group for access control time zone or not?	Bool		
14.	UseUserGroupVM	Enable the user group verification mode for access control or not?	Bool		
15.	Department	Department NO. of the device	Int		
16.	Enrolls	Array of user enrollment information	Enroll[]		
17.	AccessTimeZone	Access time zone for access control	Int		
18.	ValidDate	Valid Date	DateTime	2010-01-01 to 2099-12-31	
19.	InvalidDate	Invalid Date	DateTime	2010-01-01 to 2099-12-31	
20.	UserGroup	User group of the device	Int		
21.	LockControl	scope of control of the door lock	UInt16		

1.4. Sex

namespace：Riss.Devices

Description：User’s gender

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	Female	Gender: female	String	F	
2.	Male	Gender: male	String	M	
3.	Unknown	gender	String	Empty string	

1.5. Enroll

namespace：Riss.Devices

Description：User enrollment information

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	DIN	Device NO.	UInt64		Max. 18 digits
2.	EnrollType	Enrollment info type	EnrollType		
3.	IsDuress	Set the current enrollment info(such as current fingerprint) as the duress alarm info or not? (when people use the duress fingerprint for authentication under duress, the device will send out a silent alarm to the people inside the office.)	Bool		
4.	Fingerprint	fingerprint	Byte[]		
5.	Password	password	String		
6.	CardID	Card number	String		

1.6. EnrollExt

namespace：Riss.Devices

Description：User enrollment information, for the calls from those don't support the generic

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	DIN	Device NO.	String		Max. 18 digits
2.	EnrollType	Enrollment info type	EnrollType		
3.	IsDuress	Set the current enrollment info(such as current fingerprint) as the duress alarm info or not? (when people use the duress fingerprint for authentication under duress, the device will send out a silent alarm to the people inside the office.)	Bool		

4.	Fingerprint	fingerprint	Byte[]		
5.	Password	password	String		
6.	CardID	Card number	String		

1.7. Record

namespace：Riss.Devices

Description：Device Attendance and Records management

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	DIN	User ID in device	UInt64		Max. 18 digits
2.	DN	Device ID.	Int		
3.	Clock	Attendance time	DateTime		
4.	Verify	Verify mode	Int		
5.	Action	record of attendance details (clock-in/out, in/out, etc.)	Int		
6.	Remark	Remarks information	String		
7.	MDIN	Admin. DIN for the S.Log.(S.Log is Menu records of those who enter the Menu and change the settings, records of the Menu actions.)	UInt64		
8.	DoorStatus	Current status of the door(Reserve)	Int		
9.	JobCode	Work code	Int		
10.	Antipassback	Anti-pass back	Int		

1.8. RecordExt

namespace：Riss.Devices

Description：Device Attendance and Records management, for the calls from those don't support the generic

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	DIN	User ID in device	String		Max. 18 digits
2.	DN	Device ID.	Int		
3.	Clock	Attendance time	DateTime		
4.	Verify	Verify mode	Int		
5.	Action	record of attendance	Int		

		details (clock-in/out, in/out, etc.)			
6.	Remark	Remarks information	String		
7.	MDIN	Admin. DIN for the S.Log.(S.Log is Menu records of those who enter the Menu and change the settings, records of the Menu actions.)	String		
8.	DoorStatus	Current status of the door(Reserve)	Int		
9.	JobCode	Work code	Int		
10.	Antipassback	Anti-pass back	Int		

1.9. Monitor

namespace：Riss.Devices

Description：Real-time monitor connection class

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	Mode	Listening Mode	Int	0-1	0：UDP 1：RS485
2.	UDPAddress	IP Address	String		
3.	UDPPort	Port	Int		
4.	SerialPort	Serial Port	Int		
5.	SerialBaudRate	Baudrate	Int		

1.10. ReceiveEventArg

namespace：Riss.Devices

Description：Represents an event with no event data

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	reocrd	Device Attendance	Record		
2.	endPoint	Network endpoint	IPEndPoint		

1.11. ReceiveEventArgExt

namespace：Riss.Devices

Description：Represents an event with no event data

SN	Attribute	Attribute name	Type	Limits	Attribute description
1.	reocrd	Device Attendance	RecordExt		
2.	endPoint	Network endpoint	IPEndPoint		

Chapter 2 ENUM Definitions

2.1. CommunicationType

namespace：Riss.Devices

Description：communication type

Example：

```
public enum CommunicationType{
    Serial = 0,//serial
    Tcp,//TCP/IP
    Usb//USB
}
```

2.2. EnrollType

namespace：Riss.Devices

Description：enroll type

Example：

```
public enum EnrollType{
    /// <summary>
    /// the 1st fingerprint
    /// </summary>
    Finger0 = 0,
    Finger1,
    Finger2,
    Finger3,
    Finger4,
    Finger5,
    Finger6,
    Finger7,
    Finger8,
    /// <summary>
    /// the 10th fingerprint
    /// </summary>
    Finger9,
    Password,
    Card,
    AllFinger,
    All = 31,
}
```

Remarks：when use this ENUM in ZD2911, need to convert ENUM to Int. the value of this ENUM is 12 digits in length. Each number can be 0 or 1 only, 0: no registration; 1: registered.

NO.0 – 9th digit: NO.0 - 9th fingerprint enrollment status

No.10th digit: Password enrollment status

NO.11th digit: Card enrollment status

2.3. UserProperty

namespace：Riss.Devices

Description：settable user property

Example：

```
public enum UserProperty{
    /// <summary>
    /// Name
    /// </summary>
    UserName,

    /// <summary>
    /// Enroll
    /// </summary>
    Enroll,

    /// <summary>
    /// customized user extend information
    /// </summary>
    UserExtInfo,

    /// <summary>
    /// access control settings
    /// </summary>
    AccessControlSettings,

    /// <summary>
    /// messages, not supported
    /// </summary>
    Messages,

    /// <summary>
    /// user privilege
```

```
        /// </summary>
        Privilege,

        /// <summary>
        /// attendance rule
        /// </summary>
        Attendance,

        /// <summary>
        /// user enroll
        /// </summary>
        UserEnroll,
    }
```

2.4. UserEnrollCommand

namespace：Riss.Devices

Description：settable user enrollment information

Example：

```
public enum UserEnrollCommand {
        /// <summary>
        /// read the fingerprint data
        /// </summary>
        ReadFingerprint,

        /// <summary>
        /// write the fingerprint data
        /// </summary>
        WriteFingerprint,

        /// <summary>
        /// delete fingerprint data
        /// </summary>
        ClearFingerprint,

        /// <summary>
        /// read password
        /// </summary>
        ReadPassword,

        /// <summary>
        /// write password
        /// </summary>
```

```
        WritePassword,

        /// <summary>
        /// delete password
        /// </summary>
        ClearPassword,

        /// <summary>
        /// read card number
        /// </summary>
        ReadCard,

        /// <summary>
        /// write card number
        /// </summary>
        WriteCard,

        /// <summary>
        /// delete card number
        /// </summary>
        ClearCard,
    }
```

2.5. DeviceProperty

namespace：Riss.Devices

Description：settable device property

Example：

```
    public enum DeviceProperty
    {
        /// <summary>
        /// Firmware version
        /// </summary>
        FirmwareVersion,

        /// <summary>
        /// Firmware upgrade
        /// </summary>
        FirmwareUpgrade,

        /// <summary>
        /// built-in bell
        /// </summary>
```

```
Bell,

/// <summary>
/// door control
/// </summary>
DoorControl,

/// <summary>
/// access control settings
/// </summary>
AccessControlSettings,

/// <summary>
/// Welcome title(Home page)
/// </summary>
WelcomeTitle,

/// <summary>
/// standby title (standby page)
/// </summary>
StandbyTitle,

/// <summary>
/// initialization settng
/// </summary>
InitSettings,

/// <summary>
/// device status
/// </summary>
Status,

/// <summary>
/// power
/// </summary>
PowerOff,

/// <summary>
/// device time
/// </summary>
DeviceTime,

/// <summary>
/// get or delete the S.Log(that is management records/ records of menu actions)
```

```
from the device
        /// </summary>
        ManagementRecords,

        /// <summary>
        /// get or delete attendance records from device
        /// </summary>
        AttRecords,

        /// <summary>
        /// get or delete the list of all enroll info. (not including fingerprint data, password
and card number.)
        /// </summary>
        Enrolls,

        /// <summary>
        /// messages
        /// </summary>
        Message,

        /// <summary>
        /// device status setting: 1=busy; 0=idle
        /// </summary>
        Enable,

        /// <summary>
        /// get the count of attendance records.
        /// </summary>
        AttRecordsCount,

        /// <summary>
        /// get the count of admin records(S.Log =    management records/ records of menu
actions)
        /// </summary>
        ManagementRecordsCount,

            /// <summary>
            /// scheduled power on / shutdown
            /// </summary>
            PowerOnOffTime,

            /// <summary>
            /// Device MAC address
            /// </summary>
```

```
        MacAddress,

        /// <summary>
        /// Attendance rule in device
        /// </summary>
        Attendance,

        /// <summary>
        /// device model
        /// </summary>
        Model,

        /// <summary>
        /// Device system parameter
        /// </summary>
        SysParam,

        /// <summary>
        /// upload sound
        /// </summary>
        UploadSound,
    }
```

2.6. AccessContorlCommand

namespace：Riss.Devices

Description：settable access control property

Example：

```
public enum AccessContorlCommand
    {
        /// <summary>
        /// Pass time
        /// </summary>
        PassTime,

        /// <summary>
        /// group time for access
        /// </summary>
        GroupTime,

        /// <summary>
        /// timezone for access
        /// </summary>
```

```
    TimeZone,

    /// <summary>
    /// lock group
    /// </summary>
    LockGroup,

    /// <summary>
    /// access control parameter property in device menu (ACS → D-Lock)
    /// </summary>
    DoorKey,

    /// <summary>
    /// monitoring for attendance log
    /// </summary>
    LogWatch,

    /// <summary>
    /// user access contorl
    /// </summary>
    UserAccessCtrl,

    /// <summary>
    /// user validity period
    /// </summary>
    UserPeriod,
}
```

2.7. AttendanceCommand

namespace：Riss.Devices

Description：settable attendance time setting in device MENU (LogData → AT-Time)

Example：

```
public enum AttendanceCommand {
        /// <summary>
        /// attendance time segment
        /// </summary>
        TimeSegment,

        /// <summary>
        /// attendance time zone
        /// </summary>
        TimeZone,
```

```
        /// <summary>
        /// holiday
        /// </summary>
        Holiday,

        /// <summary>
        /// attendance time setting in device MENU(LogData → At-Time → LogTime)
        /// </summary>
        LogTime,
    }
```

2.8. NumberType

namespace：Riss.Devices

Description：Integer data type

Example：

```
public enum NumberType {
        /// <summary>
        /// UInt16
        /// </summary>
        UInt16Bit,

        /// <summary>
        /// Int16Bit
        /// </summary>
        Int16Bit,

        /// <summary>
        /// UInt32
        /// </summary>
        UInt32Bit,

        /// <summary>
```

```
        /// Int32
        /// </summary>
        Int32Bit,

        /// <summary>
        /// UInt64
        /// </summary>
        UInt64Bit,

        /// <summary>
        /// Int64
        /// </summary>
        Int64Bit,
    }
```

Chapter 3 ZD2911 Utility class Zd2911Utils

namespace：Riss.Devices

Description：provide the public external method

3.1. Constant Definitions

```
/// <summary>
///attendance rule：group count of attendance time zone
/// </summary>
public const int MaxAttendTimeZoneCount = 3;

/// <summary>
///attendance rule：group count of holiday
/// </summary>
public const int MaxHolidayCount = 30;

/// <summary>
///attendance rule：LogData – At Time → LogTime → Normal At Time → NO. count.
/// </summary>
public const int MaxValidAttendTimeCount = 24;

/// <summary>
///length of fingerprint data
/// </summary>
public const int MaxFingerprintLength = 498;

/// <summary>
/// count of fingerprint
/// </summary>
public const int MaxFingerprintCount = 10;

/// <summary>
///group count of access group time
/// </summary>
public const int GroupTimeCount = 30;

/// <summary>
///settable access control time zone count (MENU → ACS → T-Period NO.)
/// </summary>
public const int TimeZoneCount = 30;
```

```
/// <summary>
///settable access control week count (MENU → ACS → T-period Week NO.(Monday, Tuesday,..Sunday.))
/// </summary>
public const int TimeZoneWeekCount = 7;

/// <summary>
///Home title
/// </summary>
public const int DeviceTile = 60;

/// <summary>
/// standby title (standby page)
/// </summary>
public const int DeviceStandbyTitle = 61;

/// <summary>
///Model
/// </summary>
public const int DeviceModel = 62;

/// <summary>
///Firmware version
/// </summary>
public const int DeviceFirmwareVersion = 63;

/// <summary>
///Message
/// </summary>
public const int DeviceMessage = 64;

/// <summary>
///Bell
/// </summary>
public const int DeviceAlarmClock = 65;

/// <summary>
///scheduled power on / shutdown
/// </summary>
public const int DevicePowerTimer = 66;

/// <summary>
/// group count of Bell
/// </summary>
```

```
public const int BellGroupCount = 24;

/// <summary>
///length of single bell group
/// </summary>
public const int BellLength = 8;

/// <summary>
/// group count of power on/shutdown
/// </summary>
public const int PowerTimeCount = 12;

/// <summary>
///length of each power on/ shutdown group
/// </summary>
public const int PowerTimeLength = 4;

/// <summary>
///length of message
/// </summary>
public const int MaxDeviceMessageLength = 84;

/// <summary>
/// group count of message
/// </summary>
public const int MaxDeviceMessageCount = 10;

/// <summary>
/// Device communication status: Busy
/// </summary>
public const int DeviceBusy = 1;

/// <summary>
/// Device communication status: Idle
/// </summary>
public const int DeviceIdle = 0;
```

3.2. Business Methods

3.2.1. public static int BitCheck(int num, int index)

Description：Check if there's the user enrollment info (fingerprint, password, card number) in Index.

Parameter：num. = enrollment status；index = index，NO.0-9th = NO.0-9th fingerprint, 10th as password, 11th as card number.

Return Parameter：When it’s not 0, it means there’s the enrollment data in the current Index.

3.2.2. public static int SetBit(int num, int index)

Description：merge the user enrollment status

Parameter：num = 0；index，NO.0-9th = NO.0-9th fingerprint, 10th as password, 11th as card number.

Return Parameter：user enrollment information

3.2.3. public static byte[] CreateChunkHeader(byte[] buffer, int dataChunk)

Description：get the WAVE format audio file header.

Parameter：buffer WAVE；dataChunk

Return Parameter：WAVE format audio file header byte array

Chapter 4　ZD2911 Tools class Zd2911Tools

Namespace：Riss.Devices

Description：provide the public external method by COM, extends interface IZd2911Tools

4.1. public int BitCheck(int num, int index)

Description：Check if there's the user enrollment info (fingerprint, password, card number) in Index.

Parameter：num. = enrollment status；index = index，NO.0-9th = NO.0-9th fingerprint, 10th as password, 11th as card number.

Return Parameter：When it's not 0, it means there's the enrollment data in the current Index.

4.2. public int SetBit(int num, int index)

Description：merge the user enrollment status

Parameter：num = 0；index，NO.0-9th = NO.0-9th fingerprint, 10th as password, 11th as card number.

Return Parameter：user enrollment information

4.3. public byte[] CreateChunkHeader(byte[] buffer, int dataChunk)

Description：get the WAVE format audio file header.

Parameter：buffer WAVE；dataChunk

Return Parameter：WAVE format audio file header byte array

4.4. public RecordExt[] GetSLogList(ref byte[] buffer)

Description：Converter byte array to SuperLog array.

Parameter：byte array

Return Parameter：SuperLog array

4.5. public UserExt GetUserExtWithoutEnroll(ref byte[] buffer)

Description：Converter byte array to UserExt, not including the fingerprint data, password and

card NO.

Parameter：byte array

Return Parameter：UserExt

4.6. public UserExt[] GetAllUserExtWithoutEnroll(ref byte[] buffer)

Description：Converter byte array to UserExt array, not including the fingerprint data, password and card NO.

Parameter：byte array

Return Parameter：UserExt array

4.7. public RecordExt[] GetGLogList(ref byte[] buffer)

Description：Converter byte array to GeneralLog array.

Parameter：byte array

Return Parameter：GeneralLog array

4.8. public string GetString(ref byte[] bs)

Description：Decodes all the bytes in the specified byte array into a string by unicode.

Parameter：byte array

Return Parameter：A System.String containing the results of decoding the specified sequence of bytes.

4.9. public byte[] GetBytes(string input)

Description：Encodes all the characters in the specified System.String into a sequence of bytes by unicode.

Parameter：string

Return Parameter：A byte array containing the results of encoding the specified set of characters.

4.10. public string GetStringByNum(ref byte[] bs, int index, NumberType type)

Description：Converter byte array to string.

Parameter：bs = byte array; index = index; type = Integer data type

Return Parameter：A string representation of the integer

4.11. public byte[] GetBytesByNum(string input, NumberType type)

Description：Converter string to byte array.

Parameter：input= A string representation of the integer; type = Integer data type

Return Parameter：byte array

4.12. public string GetASCII(ref byte[] bs, int index, int length)

Description：Decodes a sequence of bytes from the specified byte array into a string by ASCII.

Parameter：bs= The byte array containing the sequence of bytes to decode; index= The index of the first byte to decode; length = The number of bytes to decode

Return Parameter：A System.String containing the results of decoding the specified sequence of bytes.

4.13. public int ConvertIPAddressToNumber(string strIPAddress)

Description：Converter IPAddress to numeric.

Parameter：strIPAddress= IPAddress, example: 192.168.1.181

Return Parameter：int.

4.14. public string ConvertNumberToIPAddress(int intIPAddress)

Description：Converter numeric to IPAddress.

Parameter：int

Return Parameter：string

Chapter 5 ZD2911 File Management Class

Zd2911EnrollFileManagement

Namespace：Riss.Devices

Description：save the user enrollment data to file or open the user enrollment data from file

5.1. public bool SaveAllUserEnrollDataAsDB(string fileName, List<User> userList)

Description：save all the user enrollment data to file

Parameter：fileName ； userList = collection of the user information

Return Parameter：Bool

5.2. public bool SaveUserEnrollDataAsDB(string fileName, User user)

Description：save a single user enrollment data to file

Parameter：fileName ； user =user entities

Return Parameter：Bool

5.3. public bool LoadAllUserEnrollDataFromDB(string fileName, ref List<User> userList)

Description：read all the user enrollment data from file

Parameter：fileName；userList = return the user information that read from file.

Return Parameter：Bool

5.4. public bool LoadUserEnrollDataFromDB(string fileName, ref User user)

Description：read a single user enrollment data from file

Parameter：fileName；user = return the single user information that read from file.

Return Parameter：Bool

5.5. public bool SaveUserNameData(string fileName, List<User> userList)

Description：save the user name to file

Parameter：fileName；userList = collection of user information

Return Parameter：Bool

5.6. public bool LoadUserNameData(string fileName, ref List<User> userList)

Description：read the user information(including user ID and user name)from file.

Parameter：fileName；userList = return the collection of user information(including user ID and user name)that read from file.

Chapter 6 ZD2911 File Management Class

Zd2911EnrollFile

Namespace：Riss.Devices

Description：save the user enrollment data to file or open the user enrollment data from file, for the calls from those don't support the generic, extends interface IZd2911EnrollFile

6.1. public bool SaveAllUserExtEnrollDataAsDB(string fileName, ref UserExt[] userExts)

Description：save all the user enrollment data to file, for the calls from those don't support the generic

Parameter：fileName ； userExts = array of the user information

Return Parameter：Bool

6.2. public bool SaveUserExtEnrollDataAsDB(string fileName, ref UserExt userExt)

Description：save a single user enrollment data to file, for the calls from those don't support the generic

Parameter：fileName ； userExt =user entities

Return Parameter：Bool

6.3. public bool LoadAllUserExtEnrollDataFromDB(string fileName, ref UserExt[] userExts)

Description：read all the user enrollment data from file, for the calls from those don't support the generic

Parameter：fileName；userExts = return the user information that read from file.

Return Parameter：Bool

6.4. public bool LoadUserExtEnrollDataFromDB(string fileName, ref UserExt userExt)

Description：read a single user enrollment data from file, for the calls from those don't support the generic

Parameter：fileName；userExt = return the single user information that read from file.

Return Parameter：Bool

6.5. public bool SaveUserExtNameData(string fileName, ref UserExt[] userExts)

Description：save the user name to file, for the calls from those don't support the generic

Parameter：fileName；userExts = array of user information

Return Parameter：Bool

6.6. public bool LoadUserExtNameData(string fileName, ref UserExt[] userExts)

Description：read the user information(including user ID and user name)from file, for the calls from those don't support the generic.

Parameter：fileName；userExts = return the array of user information(including user ID and user name)that read from file.

Chapter 7　Real-time monitoring Class Zd2911Monitor

Namespace：Riss.Devices

Description：Listen the attendance records that are uploaded from device in real-time. Supports two listening mode: UDP, RS485.

7.1 public static Zd2911Monitor CreateZd2911Monitor(Monitor m)

Description：Create Zd2911Monitor

Parameter: Monitor

Return Parameter: Zd2911Monitor

7.2 public bool OpenListen()

Description：Open the listener

Return Parameter：bool

7.3 public void CloseListen()

Description：Close the Listener

7.4 public event ReceiveHandler ReceiveHandler

Description：in response to an event, receive the attendance records that are uploaded from device in real-time.

7.5 public bool IsBusy

Description：true: A monitor is busy.

Chapter 8 Real-time monitoring Class Zd2911Listener

Namespace：Riss.Devices

Description：Listen the attendance records that are uploaded from device in real-time.

Supports two listening mode: UDP, RS485, extends interface IZd2911Listener.

8.1 public bool OpenListen(Monitor m)

Description：Open the listener

Parameter: Monitor

Return Parameter：bool

8.2 public void CloseListen()

Description：Close the Listener

8.3 public event ReceiveHandlerExt ReceiveHandlerExt

Description：in response to an event, receive the attendance records that are uploaded from device in real-time.

8.4 public bool IsBusy

Description：true: A monitor is busy.

Chapter 9 Description for the device communication interface

ClassName：DeviceConnection

Namespace：Riss.Devices

Description：device connection base class

9.1. public static DeviceConnection CreateConnection(ref Device device)

Description：Create the device connection

Parameter：Device Entity

Return Parameter：DeviceConnection

9.2. public abstract int Open()

Description：open the connection with the device, when the return parameter is greater than 0, it means connection succeful.

Return Parameter：Int

9.3. public abstract void Close()

Description：Close the connection with the device

9.4. public abstract bool SetProperty(UserProperty property, object extraProperty, User user, object extraData)

Description：set the user related informtionin device

Parameter：property = settable user property; extraProperty = settable extended property; user= User entity; extraData = extra information

Return Parameter：Bool

9.5. public abstract bool GetProperty(UserProperty property, object extraProperty, ref User user, ref object extraData)

Description：Get the user related information from device.

Parameter：property = settable user property; extraProperty= settable extended property;

user= User entity; extraData = extra information

Return Parameter：Bool

9.6. public abstract bool SetProperty(DeviceProperty property, object extraProperty, Device device, object extraData)

Description：set the information that related to the device

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

9.7. public abstract bool GetProperty(DeviceProperty property, object extraProperty, ref Device device, ref object extraData)

Description：Get the information that related to the device.

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

9.8. public bool SetPropertyExt(UserProperty property, ref byte[] extraProperty, UserExt user, ref byte[] extraData)

Description：set the user related informtionin device, for the calls from those don't support the generic

Parameter：property = settable user property; extraProperty = settable extended property; user= UserExt entity; extraData = extra information

Return Parameter：Bool

9.9. public bool GetPropertyExt(UserProperty property, ref byte[] extraProperty, ref UserExt user, ref byte[] extraData)

Description：Get the user related information from device, for the calls from those don't support the generic

Parameter：property = settable user property; extraProperty= settable extended property; user= User entity; extraData = extra information

Return Parameter：Bool

9.10. public bool SetPropertyExt(DeviceProperty property, ref byte[] extraProperty, Device device, ref byte[] extraData)

Description：Get the information that related to the device, for the calls from those don't support the generic

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

9.11. public bool GetPropertyExt(DeviceProperty property, ref byte[] extraProperty, ref Device device, ref byte[] extraData)

Description：Get the information that related to the device, for the calls from those don't support the generic

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

Chapter 10 Device Connection Class

Zd2911DeviceConnection

Namespace：Riss.Devices

Description：device connection class, for the calls from those don't support the generic(COM), extends interface IZd2911DeviceConnection

10.1. public int Open(Device device)

Description：open the connection with the device, when the return parameter is greater than 0, it means connection succeful.

Parameter: Device entity

Return Parameter：Int

10.2. public void Close()

Description：Close the connection with the device

10.3. public bool SetPropertyExt(UserProperty property, ref byte[] extraProperty, UserExt user, ref byte[] extraData)

Description：set the user related informtionin device, for the calls from those don't support the generic

Parameter：property = settable user property; extraProperty = settable extended property; user= UserExt entity; extraData = extra information

Return Parameter：Bool

10.4. public bool GetPropertyExt(UserProperty property, ref byte[] extraProperty, ref UserExt user, ref byte[] extraData)

Description：Get the user related information from device, for the calls from those don't support the generic

Parameter：property = settable user property; extraProperty= settable extended property; user= User entity; extraData = extra information

Return Parameter：Bool

10.5. public bool SetPropertyExt(DeviceProperty property, ref byte[] extraProperty, Device device, ref byte[] extraData)

Description：Get the information that related to the device, for the calls from those don't support the generic

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

10.6. public bool GetPropertyExt(DeviceProperty property, ref byte[] extraProperty, ref Device device, ref byte[] extraData)

Description：Get the information that related to the device, for the calls from those don't support the generic

Parameter：property = settable device property; extraProperty = settable extended property; device = Device entity； extraData = extra information

Return Parameter：Bool

Chapter 11 Settable User Information in Device

11.1. SetUserName

Function: SetProperty(UserProperty.UserName, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
user.UserName = "Amaris";
bool result = deviceConnection.SetProperty(UserProperty.UserName, extraProperty, user, extraData);

11.2. SetUserExtInfo

Function: SetProperty(UserProperty.UserExtInfo, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
user.Comment = "User extended information";
bool result = deviceConnection.SetProperty(UserProperty.UserExtInfo, extraProperty, user, extraData);

11.3. SetUserRole

Function: SetProperty(UserProperty.Privilege, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
user.Privilege = 1;
bool result = deviceConnection.SetProperty(UserProperty.Privilege, extraProperty, user, extraData);

11.4. SetUserAccess

Function: SetProperty(UserProperty.AccessControlSettings, extraProperty, user, extraData)

Remark: Belong to the access control settings option

Example

object extraProperty = new object();
extraProperty = AccessContorlCommand.UserAccessCtrl;
object extraData = new object();
User user = new User();
user.DIN = 1;
user.AccessTimeZone = 1;
user.Enable = true;
bool result = deviceConnection.SetProperty(UserProperty.AccessControlSettings, extraProperty, user, extraData);

11.5. SetUserPeriod

Function: SetProperty(UserProperty.AccessControlSettings, extraProperty, user, extraData)

Remark: Belong to the access control settings option

Example

object extraProperty = new object();
extraProperty = AccessContorlCommand.UserPeriod;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
byte[] data = new byte[8];
data[0] = (byte)(2012 - 2000);//Start Date: Year
data[1] = (byte)8;//Start Date: Month
data[2] = (byte)8;//Start Date: Day
data[3] = (byte)(2018 - 2000);//End Date: Year
data[4] = (byte)12;//End Date: Month
data[5] = (byte)31;//End Date: Day
extraData = data;
bool result = deviceConnection.SetProperty(UserProperty.AccessControlSettings, extraProperty, user, extraData);

11.6. SetUserEnrollData

Function: SetProperty(UserProperty.Enroll, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
extraData = false;
result = deviceConnection.SetProperty(UserProperty.Enroll, extraProperty, user, extraData);

11.7. SetUserAtType

Function: SetProperty(UserProperty.Attendance, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
extraData = 1;
User user = new User();
user.DIN = (UInt64)1;
bool result = deviceConnection.SetProperty(UserProperty.Attendance, extraProperty, user, extraData);

11.8. ClearCard

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

object extraProperty = new object();
extraProperty = UserEnrollCommand.ClearCard;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
user.Enrolls.Add(enroll);
bool result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);

11.9. ClearFingerprintData

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
Enroll enroll = new Enroll();
user.DIN = (UInt64)1;
enroll.DIN = user.DIN;
enroll.EnrollType = (EnrollType)0;//FP No.
user.Enrolls.Add(enroll);
extraProperty = UserEnrollCommand.ClearFingerprint;
result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);

11.10. ClearPassword

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

object extraProperty = new object();
extraProperty = UserEnrollCommand.ClearPassword;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
user.Enrolls.Add(enroll);
bool result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);

11.11. WriteCard

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

object extraProperty = new object();
extraProperty = UserEnrollCommand.WriteCard;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
user.Privilege = 1;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
enroll.CardID = "12345678";
user.Enrolls.Add(enroll);
bool result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);

11.12. WriteFingerprintData

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

User user = new User();
Enroll enroll = new Enroll();
user.DIN = (UInt64)1;
user.Privilege = 1;
enroll.DIN = user.DIN;
enroll.EnrollType = (EnrollType)0;//FP No.
byte[] fingerprint = new byte[Zd2911Utils.MaxFingerprintLength * (cbo_FpNo.SelectedIndex + 1)];//cbo_FpNo:FP No. ComboBox

```
Array.Copy(fpBytes, 0, fingerprint, cbo_FpNo.SelectedIndex * Zd2911Utils.MaxFingerprintLength, fpBytes.Length);//fpBytes:Store fingerprint data
enroll.Fingerprint = fingerprint;
user.Enrolls.Add(enroll);
extraProperty = UserEnrollCommand.WriteFingerprint;
result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);
```

11.13. WritePassword

Function: SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData)

Example

```
object extraProperty = new object();
extraProperty = UserEnrollCommand.WritePassword;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
user.Privilege = 1;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
enroll.Password = "123123";
user.Enrolls.Add(enroll);
bool result = deviceConnection.SetProperty(UserProperty.UserEnroll, extraProperty, user, extraData);
```

Chapter 12 Obtainable User Information in Device

12.1. GetUserName

Function: GetProperty(UserProperty.UserName, extraProperty, ref user, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
bool result = deviceConnection.GetProperty(UserProperty.UserName, extraProperty, ref user, ref extraData);
if (result) {
txt_UserName.Text = user.UserName;
}

12.2. GetUserExtInfo

Function: GetProperty(UserProperty.UserExtInfo, extraProperty, ref user, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
bool result = deviceConnection.GetProperty(UserProperty.UserExtInfo, extraProperty, ref user, ref extraData);
if (result) {
ExtInfoTextBox.Text = user.Comment;
}

12.3. GetUserAccess

Function: GetProperty(UserProperty.AccessControlSettings, extraProperty, ref user, ref extraData)

Remark: Belong to the access control settings option

Example

object extraProperty = new object();
extraProperty = AccessContorlCommand.UserAccessCtrl;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;

```
bool result = deviceConnection.GetProperty(UserProperty.AccessControlSettings, extraProperty, ref user, ref extraData);
if (result) {
    int AccessTimeZone = user.AccessTimeZone;
    bool Enable = user.Enable;
}
```

12.4. GetUserPeriod

Function: GetProperty(UserProperty.AccessControlSettings, extraProperty, ref user, ref extraData)

Remark: Belong to the access control settings option

Example

```
object extraProperty = new object();
extraProperty = AccessContorlCommand.UserPeriod;
object extraData = new object();
User user=new User();
user.DIN=(UInt64)nud_PeriodDIN.Value;
bool result = deviceConnection.GetProperty(UserProperty.AccessControlSettings, extraProperty, ref user, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 8
    int startYear = data[0] + 2000;
    int startMonth = datat[1];
    int startDay = data[2];
    int endYear = data[3] + 2000;
    int endMonth = data[4];
    int endDay = data[5];
}
```

12.5. GetUserEnrollData

Function: GetProperty(UserProperty.Enroll, extraProperty, ref user, ref extraData)

Before call this Function, need to call the Function "deviceConnection.GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref extraData)" to get the user information collection.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = (UInt64)0;
result = deviceConnection.GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref
```

```
extraData);
if (false == result) {
    return;
}

List<User> userList = (List<User>)extraData;
foreach (User user in userList) {
    result = deviceConnection.GetProperty(UserProperty.Enroll, extraProperty, ref user, ref extraData);
}
```

12.6. GetUserAtType

Function: GetProperty(UserProperty.Attendance, extraProperty, ref user, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
bool result = deviceConnection.GetProperty(UserProperty.Attendance, extraProperty, ref user, ref extraData);
if (result) {
    int userAttendType = (int)extraData;
}
```

12.7. ReadCard

Function: GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData)

Example

```
object extraProperty = new object();
extraProperty = UserEnrollCommand.ReadCard;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)1;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
user.Enrolls.Add(enroll);
bool result = deviceConnection.GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData);
if (result) {
    txt_Card.Text = user.Enrolls[0].CardID;
}
```

12.8. ReadFingerprintData

Function: GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
User user = new User();
Enroll enroll = new Enroll();
user.DIN = (UInt64)1;
enroll.DIN = user.DIN;
enroll.EnrollType = (EnrollType)cbo_FpNo.SelectedIndex;//FP No.
enroll.Fingerprint = new byte[Zd2911Utils.MaxFingerprintLength];
user.Enrolls.Add(enroll);
extraProperty = UserEnrollCommand.ReadFingerprint;
result = deviceConnection.GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData);
if (result) {
fpBytes = user.Enrolls[0].Fingerprint;
}

12.9. ReadPassword

Function: GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData)

Example

object extraProperty = new object();
extraProperty = UserEnrollCommand.ReadPassword;
object extraData = new object();
User user = new User();
user.DIN = (UInt64)nud_DIN.Value;
Enroll enroll = new Enroll();
enroll.DIN = user.DIN;
user.Enrolls.Add(enroll);
bool result = deviceConnection.GetProperty(UserProperty.UserEnroll, extraProperty, ref user, ref extraData);
if (result) {
txt_Pwd.Text = user.Enrolls[0].Password.Replace("\0","");
}

Chapter 13 Settable Information in Device

13.1. SetGroupTime

Function: SetProperty(DeviceProperty.AccessControlSettings, extraProperty, device, extraData)

Remark: set 30 groups of data at one time(the length of the byte array is 30*10), each set of data is an array of 10 bytes in size. Belong to the access control settings option.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AccessContorlCommand.GroupTime;
result = deviceConnection.GetProperty(DeviceProperty.AccessControlSettings, extraProperty, ref device, ref extraData);//Get device  GroupTime Settings
if (false == result) {
    return;
}
byte[] data = (byte[])extraData;
data[cbo_GroupNo.SelectedIndex * 10 + 2] = (byte)(1);
data[cbo_GroupNo.SelectedIndex * 10 + 3] = (byte)(1);
data[cbo_GroupNo.SelectedIndex * 10 + 4] = (byte)(1);
data[cbo_GroupNo.SelectedIndex * 10 + 5] = (byte)(1);
extraData = data;
result = deviceConnection.SetProperty(DeviceProperty.AccessControlSettings, extraProperty, device, extraData);
```

13.2. SetTimeZone

Function: SetProperty(DeviceProperty.AccessControlSettings, extraProperty, device, extraData)

Remark: set 30*7 groups of data at one time(the length of the byte array is 30*7*6), each set of data is an array of 6 bytes in size. Belong to the access control settings option.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AccessContorlCommand.TimeZone;
result = deviceConnection.GetProperty(DeviceProperty.AccessControlSettings, extraProperty,
```

```
ref device, ref extraData);
if (false == result) {
      return;
}
byte[] data = (byte[])extraData;// Get the 30*7 groups of data
//Get the corresponding position data according to the time interval NO. and Week NO.
int index = 6 * (cbo_ZoneNo.SelectedIndex * Zd2911Utils.TimeZoneWeekCount + cbo_Weekday.SelectedIndex);
data[index + 2] = (byte)cbo_ZoneBeginHour.SelectedIndex;
data[index + 3] = (byte)cbo_ZoneBeginMinute.SelectedIndex;
data[index + 4] = (byte)cbo_ZoneEndHour.SelectedIndex;
data[index + 5] = (byte)cbo_ZoneEndMinute.SelectedIndex;
extraData = data;
result = deviceConnection.SetProperty(DeviceProperty.AccessControlSettings, extraProperty, device, extraData);
```

13.3. SetSysParam

Function: SetProperty(DeviceProperty.SysParam, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
byte[] data = new byte[8];
Array.Copy(BitConverter.GetBytes(1), 0, data, 0, 4);//Param Index
Array.Copy(BitConverter.GetBytes(1), 0, data, 4, 4);//Param Value
extraData = data;
bool result = deviceConnection.SetProperty(DeviceProperty.SysParam, extraProperty, device, extraData);
```

13.4. InitDevice

Function: SetProperty(DeviceProperty.InitSettings, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
result = deviceConnection.SetProperty(DeviceProperty.InitSettings, extraProperty, device, extraData);
```

13.5. EnableDevice

Function: SetProperty(DeviceProperty.Enable, extraProperty, device, extraData)

Reamrk: 1 Device Busy; 0 Device Idle

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = (long)1;
result = deviceConnection.SetProperty(DeviceProperty.InitSettings, extraProperty, device, extraData);
```

13.6. SetDeviceTime

Function: SetProperty(DeviceProperty.DeviceTime, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
bool result = deviceConnection.SetProperty(DeviceProperty.DeviceTime, extraProperty, device, extraData);
```

13.7. SetWelcomeTitle

Function: SetProperty(DeviceProperty.WelcomeTitle, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
extraProperty = Zd2911Utils.DeviceTile;
object extraData = new object();
extraData = "Welcome";
bool result = deviceConnection.SetProperty(DeviceProperty.WelcomeTitle, extraProperty, device, extraData);
```

13.8. SetStandbyTitle

Function: SetProperty(DeviceProperty.StandbyTitle, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
extraProperty = Zd2911Utils.DeviceStandbyTitle;
object extraData = new object();
extraData = "Press any key to wake up the device";
bool result = deviceConnection.SetProperty(DeviceProperty.StandbyTitle, extraProperty, device, extraData);
```

13.9. SetMessage

Function: SetProperty(DeviceProperty.Message, extraProperty, device, extraData)

Remark: set 10 groups of data at one time (the length of the byte array is 10*84), each set of data is an array of 84 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceMessage;
result = deviceConnection.GetProperty(DeviceProperty.Message, extraProperty, ref device, ref extraData);
if (false == result) {
    return;
}

byte[] data = Encoding.Unicode.GetBytes((string)extraData);//All Message Settings From Device, data.Length = 10 * 84, Message Group: 10, One Message Length: 84 byte
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 0] =
                    Convert.ToByte(chk_MessageEnable.Checked);//Enable
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 1] =
                    (byte)cbo_MessageType.SelectedIndex;//Message Type
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 2] =
                    (byte)cbo_MessageSound.SelectedIndex;//Sound
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 3] = 0;
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 4] = 0;
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 5] = 0;
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 6] =
                    (byte)(dtp_MessageBeginDatetime.Value.Year - 2000);//Start Date: Year
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 7] =
                    (byte)dtp_MessageBeginDatetime.Value.Month;//Start Date: Month
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 8] =
                    (byte)dtp_MessageBeginDatetime.Value.Day;//Start Date: Day
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 9] =
                    (byte)dtp_MessageBeginDatetime.Value.Hour;//Start Time: Hour
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 10] =
                    (byte)dtp_MessageBeginDatetime.Value.Minute;//Start Time: Minute
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 11] =
                    (byte)(dtp_MessageEndDatetime.Value.Year - 2000);//End Date: Year
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 12] =
                    (byte)dtp_MessageEndDatetime.Value.Month;//End Date: Month
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 13] =
                    (byte)dtp_MessageEndDatetime.Value.Day;//End Date: Day
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 14] =
                    (byte)dtp_MessageEndDatetime.Value.Hour;//End Time: Hour
data[cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 15] =
                    (byte)dtp_MessageEndDatetime.Value.Minute;//End Time: Minute
byte[] IDBytes = BitConverter.GetBytes((UInt64)nud_MessageID.Value);//ID
Array.Copy(IDBytes, 0, data, cbo_MessageSN.SelectedIndex *
```

```
Zd2911Utils.MaxDeviceMessageLength + 16,
                        IDBytes.Length);
byte[] contentBytes = Encoding.Unicode.GetBytes(txt_MessageContent.Text.Trim());//Message Content
Array.Copy(contentBytes, 0, data, cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength + 24,
                        contentBytes.Length);
extraData = Encoding.Unicode.GetString(data);
extraProperty = Zd2911Utils.DeviceMessage;
result = deviceConnection.SetProperty(DeviceProperty.Message, extraProperty, device, extraData);
```

13.10. Set Bell

Function: SetProperty(DeviceProperty.Bell, extraProperty, device, extraData)

Remark: set 24 groups of data at one time (the length of the byte array is 24*8), each set of data is an array of 8 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceAlarmClock;
result = deviceConnection.GetProperty(DeviceProperty.Bell, extraProperty, ref device, ref extraData);
if (false == result) {
    return;
}

string bellData = (string)extraData;
byte[] data = Encoding.Unicode.GetBytes(bellData);
//set the corresponding position data according to the NO.
data[cbo_AlarmDN.SelectedIndex * Zd2911Utils.BellLength + 0] = (byte)cbo_AlarmHour.SelectedIndex;//Hour
data[cbo_AlarmDN.SelectedIndex * Zd2911Utils.BellLength + 1] = (byte)cbo_AlarmMinute.SelectedIndex;//Minute
data[cbo_AlarmDN.SelectedIndex * Zd2911Utils.BellLength + 2] = (byte)cbo_AlarmCycle.SelectedIndex;//Cycle
data[cbo_AlarmDN.SelectedIndex * Zd2911Utils.BellLength + 3] = (byte)nud_AlarmDelay.Value;//Delay
extraProperty = Zd2911Utils.DeviceAlarmClock;
extraData = Encoding.Unicode.GetString(data);
result = deviceConnection.SetProperty(DeviceProperty.Bell, extraProperty, device, extraData);
```

13.11. SetPowerOnOffTime

Function: SetProperty(DeviceProperty.PowerOnOffTime, extraProperty, device, extraData)

Remark: set 12*2 groups of data at one time (the length of the byte array is 12*2*4), each set of data is an array of 4 bytes in size. It's 12 groups scheduled power on data between Index NO.0 – NO.47. it's 12 groups scheduled shutdown data between Index NO.48 – NO.85.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DevicePowerTimer;
result = deviceConnection.GetProperty(DeviceProperty.PowerOnOffTime, extraProperty, ref device, ref extraData);
if (result) {
    string timerData = (string)extraData;
    //length of the Return byte data: 86
  //it's the 12 groups scheduled power on data between Index NO.0 – NO.47, length of each set is 4.
    //it's the 12 groups scheduled shutdown data between Index NO.48 – NO.85, length of each set is 4.
    byte[] data = Encoding.Unicode.GetBytes(timerData);
    int index = cbo_PowerType.SelectedIndex * Zd2911Utils.PowerTimeCount * Zd2911Utils.PowerTimeLength
                                + cbo_DN.SelectedIndex * Zd2911Utils.PowerTimeLength;
    //set the data in corresponding position according to the Index.
    data[index + 0] = (byte)cbo_PowerHour.SelectedIndex;//Hour
    data[index + 1] = (byte)cbo_PowerMinute.SelectedIndex;//Minute
    data[index + 2] = Convert.ToByte(chk_Power.Checked);//Enable
    extraProperty = Zd2911Utils.DevicePowerTimer;
    extraData = Encoding.Unicode.GetString(data);
    result = deviceConnection.SetProperty(DeviceProperty.PowerOnOffTime, extraProperty, device, extraData);
}
```

13.12. EmptySuperLogData

Function: SetProperty(DeviceProperty.ManagementRecords, extraProperty, device, extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
result = deviceConnection.SetProperty(DeviceProperty.ManagementRecords, extraProperty, device, extraData);
```

13.13. EmptyGeneralLogData

Function: SetProperty(DeviceProperty.AttRecords, extraProperty, device, extraData)

Example

object extraProperty = new object();
object extraData = new object();
result = deviceConnection. SetProperty(DeviceProperty.AttRecords, extraProperty, device, extraData);

13.14. EmptyUserEnrollInfo

Function: SetProperty(DeviceProperty.Enrolls, extraProperty, device, extraData)

Remark: DIN = 0 Clear All User; Din != 0 Clear User By DIN

Example

object extraProperty = new object();
object extraData = new object();
extraData = (UInt64)1;//User DIN
result = deviceConnection.SetProperty(DeviceProperty.Enrolls, extraProperty, device, extraData);

13.15. SetMacAddress

Function: SetProperty(DeviceProperty. MacAddress, extraProperty, device, extraData)

Example

object extraProperty = new object();
object extraData = new object();
extraData = macAddress;//byte[] macAddress = new byte[6], macAddress.Length = 6
result = deviceConnection. SetProperty(DeviceProperty. MacAddress, extraProperty, device, extraData);

13.16. SetAttendTimeZone

Function: SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData)

Remark: set 3 groups of data at one time (the length of the byte array is 3*14), each set of data is an array of 14 bytes in size.

Example
object extraProperty = new object(); object extraData = new object();

```
extraProperty = AttendanceCommand.TimeZone;
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (false == result) {
    return;
}

byte[] data = (byte[])extraData;//data.Length = 3 * 14, Serial Number: data[0],data[1]
data[cbo_AttNo.SelectedIndex * 14 + 2] = (byte)cbo_AttBeginHour.SelectedIndex;//Start Time: Hour
data[cbo_AttNo.SelectedIndex * 14 + 3] = (byte)cbo_AttBeginMinute.SelectedIndex;//Start Time: Minute
data[cbo_AttNo.SelectedIndex * 14 + 4] = (byte)cbo_AttEndHour.SelectedIndex;//End Time: Hour
data[cbo_AttNo.SelectedIndex * 14 + 5] = (byte)cbo_AttEndMinute.SelectedIndex;//End Time: Minute
data[cbo_AttNo.SelectedIndex * 14 + 6] = (byte)nud_AttTimeBeforeOne.Value;//Time 1 Left
data[cbo_AttNo.SelectedIndex * 14 + 7] = (byte)nud_AttTimeAfterOne.Value;//Time 1 Right
data[cbo_AttNo.SelectedIndex * 14 + 8] = (byte)nud_AttTimeBeforeTwo.Value;//Time 2 Left
data[cbo_AttNo.SelectedIndex * 14 + 9] = (byte)nud_AttTimeAfterTwo.Value;//Time 2 Right
data[cbo_AttNo.SelectedIndex * 14 + 10] = Convert.ToByte(chk_AttTime1Enable.Checked);//Time 1 Enable
data[cbo_AttNo.SelectedIndex * 14 + 11] = Convert.ToByte(chk_AttTime2Enable.Checked);//Time 2 Enable
data[cbo_AttNo.SelectedIndex * 14 + 12] = Convert.ToByte(chk_TimeZoneUse.Checked);//Time Zone Enable
extraData = data;
result = deviceConnection.SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData);
```

13.17. SetHoliday

Function: SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData)

Remark: set 40 groups of data at one time (the length of the byte array is 40*10), each set of data is an array of 10 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AttendanceCommand.Holiday;
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (false == result) {
    return;
```

```
}

byte[] data = (byte[])extraData;//data.Length = 40 * 10, Serail Number:data[0],data[1]
//set the corresponding position data according to the NO.
data[cbo_HolidayNo.SelectedIndex * 10 + 2] = (byte)(dtp_HolidayBeginDate.Value.Year - 2000);//Start Date: Year
data[cbo_HolidayNo.SelectedIndex * 10 + 3] = (byte)dtp_HolidayBeginDate.Value.Month;//Start Date: Month
data[cbo_HolidayNo.SelectedIndex * 10 + 4] = (byte)dtp_HolidayBeginDate.Value.Day;//Start Date: Day
data[cbo_HolidayNo.SelectedIndex * 10 + 5] = (byte)(dtp_HolidayEndDate.Value.Year - 2000);//End Date: Year
data[cbo_HolidayNo.SelectedIndex * 10 + 6] = (byte)dtp_HolidayEndDate.Value.Month;//End Date: Month
data[cbo_HolidayNo.SelectedIndex * 10 + 7] = (byte)dtp_HolidayEndDate.Value.Day;//End Date: Day
extraData = data;
result = deviceConnection.SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData);
```

13.18. SetValidAtTime

Function: SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData)

Reamrk: set 24 groups of data at one time (the length of the byte array is 24*6), each set of data is an array of 6 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AttendanceCommand.LogTime;
extraData = cbo_AttType.SelectedIndex;//Attendance Type
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (false == result) {
    return;
}

byte[] data = (byte[])extraData;//data.Length = 24 * 6, Attendance Type: data[0], Serial Number: data[1]
data[cbo_AttDN.SelectedIndex * 6 + 2] = (byte)cbo_BeginHour.SelectedIndex;//Start Time: Hour
data[cbo_AttDN.SelectedIndex * 6 + 3] = (byte)cbo_BeginMinute.SelectedIndex;//Start Time: Minute
data[cbo_AttDN.SelectedIndex * 6 + 4] = (byte)cbo_EndHour.SelectedIndex;//End Time: Hour
```

```
data[cbo_AttDN.SelectedIndex * 6 + 5] = (byte)cbo_EndMinute.SelectedIndex;//End Time: Minute
extraData = data;
result = deviceConnection.SetProperty(DeviceProperty.Attendance, extraProperty, device, extraData);
```

13.19. UploadSound

Function: SetProperty(DeviceProperty.UploadSound, extraProperty, device, extraData)

Remark: Upload Audio file supports the WAVE format only, and supports USB communication only.

Example

```
private bool UploadSound(byte[] soundBuffer) {
        int chunkHeaderLen = 0;
        byte[] chunkHeader = Zd2911Utils.CreateChunkHeader(soundBuffer, dataChunk);
        int len = BitConverter.ToInt32(chunkHeader, 4);
        int downSize = len + 4;
        byte[] downData = new byte[downSize];
        Array.Copy(chunkHeader, 4, downData, 0, 4);
        Array.Copy(soundBuffer, dataChunk + chunkHeaderLen, downData, 4, len);

        int unit = 1024 * 60, i;
        bool result = false;
        int n = (int)downSize / unit;
        object extraProperty = new object();
        object extraData = new object();

        for (i = 0; i < n; i++) {
            byte[] dataBytes = new byte[unit];
            Array.Copy(downData, i * unit, dataBytes, 0, unit);
            List<int> soundParam = new List<int>();
            soundParam.Add(cbo_SoundType.SelectedIndex + 8);
            soundParam.Add(i * unit);
            extraProperty = soundParam;
            extraData = dataBytes;
            result = deviceConnection.SetProperty(DeviceProperty.UploadSound, extraProperty, device, extraData);
            if (false == result) {
                return false;
            }
        }
```

```
                n = downSize % unit;
                if (n > 0) {
                    byte[] dataBytes = new byte[n];
                    Array.Copy(downData, i * unit, dataBytes, 0, n);
                    List<int> soundParam = new List<int>();
                    soundParam.Add(cbo_SoundType.SelectedIndex + 8);
                    soundParam.Add(i * unit);
                    extraProperty = soundParam;
                    extraData = dataBytes;
                    result = deviceConnection.SetProperty(DeviceProperty.UploadSound,
extraProperty, device, extraData);
                    if (false == result) {
                        return false;
                    }
                }

                return result;
}
```

Chapter 14 Obtainable information in Device

14.1. FirmwareVersion

Function: GetProperty(DeviceProperty.FirmwareVersion, extraProperty, ref device, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceFirmwareVersion;
bool result = deviceConnection.GetProperty(DeviceProperty.FirmwareVersion, extraProperty, ref device, ref extraData);
if (result) {
    string firmwareVersion = (string)extraData;
}
```

14.2. GetGroupTime

Function: GetProperty(DeviceProperty.AccessControlSettings, extraProperty, ref device, ref extraData)

Remark: get 30 groups of data at one time (the length of the byte array is 30*10), each set of data is an array of 10 bytes in size. Belong to the access control settings option.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AccessContorlCommand.GroupTime;
result = deviceConnection.GetProperty(DeviceProperty.AccessControlSettings, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 30 * 10
    byte[] timerGroup = new byte[10];
        Array.Copy(data, cbo_GroupNo.SelectedIndex * 10, timerGroup, 0, timerGroup.Length);//Get the corresponding position data according to the NO.
        cbo_GroupMultiUser.SelectedIndex = timerGroup[2] - 1;
        cbo_ZoneNoOne.SelectedIndex = timerGroup[3] - 1;
        cbo_ZoneNoTwo.SelectedIndex = timerGroup[4] - 1;
        cbo_ZoneNoThree.SelectedIndex = timerGroup[5] - 1;
}
```

14.3. GetTimeZone

Function: GetProperty(DeviceProperty.AccessControlSettings, extraProperty, ref device, ref extraData)

Remark: set 30*7 groups of data at one time (the length of the byte array is 30*7*6), each set of data is an array of 6 bytes in size. Belong to the access control settings option.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AccessContorlCommand.TimeZone;
result = deviceConnection.GetProperty(DeviceProperty.AccessControlSettings, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 30 * 7 * 6
        byte[] timerZone = new byte[6];
        int index = timerZone.Length * (cbo_ZoneNo.SelectedIndex * Zd2911Utils.TimeZoneWeekCount
                + cbo_Weekday.SelectedIndex);//get the corresponding position data according to the Time Interval NO. and Week NO.
        Array.Copy(data, index, timerZone, 0, timerZone.Length);
        cbo_ZoneBeginHour.SelectedIndex = timerZone[2];
        cbo_ZoneBeginMinute.SelectedIndex = timerZone[3];
        cbo_ZoneEndHour.SelectedIndex = timerZone[4];
        cbo_ZoneEndMinute.SelectedIndex = timerZone[5];
}
```

14.4. GetSysParam

Function: GetProperty(DeviceProperty.SysParam, extraProperty, ref device, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
byte[] paramIndex = BitConverter.GetBytes(1);//Param Index
extraData = paramIndex;
bool result = deviceConnection.GetProperty(DeviceProperty.SysParam, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;
}
```

14.5. GetDeviceTime

Function: GetProperty(DeviceProperty.DeviceTime, extraProperty, ref device, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
bool result = deviceConnection.GetProperty(DeviceProperty.DeviceTime, extraProperty, ref device, ref extraData);
if (result) {
DateTime dt = (DateTime)extraData;
}

14.6. GetWelcomeTitle

Function: GetProperty(DeviceProperty.WelcomeTitle, extraProperty, ref device, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceTile;
bool result = deviceConnection.GetProperty(DeviceProperty.WelcomeTitle, extraProperty, ref device, ref extraData);
if (result) {
string welcomeTitle = (string)extraData;
}

14.7. StandbyTitle

Function: GetProperty(DeviceProperty.StandbyTitle, extraProperty, ref device, ref extraData)

Example

object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceStandbyTitle;
bool result = deviceConnection.GetProperty(DeviceProperty.StandbyTitle, extraProperty, ref device, ref extraData);
if (result) {
string standbyTitle = (string)extraData;
}

14.8. GetMessage

Function: GetProperty(DeviceProperty.Message, extraProperty, ref device, ref extraData)

Remark: get 10 groups of data at one time (the length of the byte array is 10*84), each set of data is an array of 84 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceMessage;
result = deviceConnection.GetProperty(DeviceProperty.Message, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = Encoding.Unicode.GetBytes((string)extraData);
    byte[] message = new byte[Zd2911Utils.MaxDeviceMessageLength];
        Array.Copy(data, cbo_MessageSN.SelectedIndex * Zd2911Utils.MaxDeviceMessageLength,
                   message, 0, message.Length);
        chk_MessageEnable.Checked = Convert.ToBoolean(message[0]);//Enable
        cbo_MessageType.SelectedIndex = message[1];//Message Type
        cbo_MessageSound.SelectedIndex = message[2];//Sound
        dtp_MessageBeginDatetime.Value = new DateTime(message[6] + 2000, message[7], message[8], message[9],
                   message[10], 0);//Start Date
        dtp_MessageEndDatetime.Value = new DateTime(message[11] + 2000, message[12], message[13], message[14],
                   message[15], 0);//End Date
        nud_MessageID.Value = BitConverter.ToUInt64(message, 16);//User DIN
        txt_MessageContent.Text = Encoding.Unicode.GetString(message, 24, 30 * 2).Replace("\0", "");//Message Content
}
```

14.9. GetBell

Function: GetProperty(DeviceProperty.Bell, extraProperty, ref device, ref extraData)

Remark: get 24 groups of data at one time (the length of the byte array is 24*8), each set of data is an array of 8 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceAlarmClock;
```

```
result = deviceConnection.GetProperty(DeviceProperty.Bell, extraProperty, ref device, ref extraData);
if (result) {
    string bellData = (string)extraData;
        byte[] dataBytes = Encoding.Unicode.GetBytes(bellData);// length of the Return byte data：192
        byte[] bell = new byte[Zd2911Utils.BellLength];
        Array.Copy(dataBytes, cbo_AlarmDN.SelectedIndex * Zd2911Utils.BellLength,
                                bell, 0, bell.Length);// Get the corresponding position data according to the NO. length of each set: 8.
        cbo_AlarmHour.SelectedIndex = bell[0];//Hour
        cbo_AlarmMinute.SelectedIndex = bell[1];//Minute
        cbo_AlarmCycle.SelectedIndex = bell[2];//Cycle
        nud_AlarmDelay.Value = bell[3];//Delay
}
```

14.10. GetPowerOnOffTime

Function: GetProperty(DeviceProperty.PowerOnOffTime, extraProperty, ref device, ref extraData)

Remark: set 12*2 groups of data at one time (the length of the byte array is 12*2*4), each set of data is an array of 4 bytes in size. It's the 12 groups scheduled power on data between Index NO.0 – NO.47. It's the 12 groups scheduled shutdown data between Index NO.48 – NO.85.

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DevicePowerTimer;
result = deviceConnection.GetProperty(DeviceProperty.PowerOnOffTime, extraProperty, ref device, ref extraData);
if (result) {
    string timerData = (string)extraData;
        // length of the Return byte data：86
        // it's the 12 groups scheduled power on data between Index NO.0 – NO.47, length of each set is 4.
        // it's the 12 groups scheduled shutdown data between Index NO.48 – NO.85, length of each set is 4.
        byte[] data = Encoding.Unicode.GetBytes(timerData);
        byte[] timer = new byte[Zd2911Utils.PowerTimeLength];
        int index = cbo_PowerType.SelectedIndex * Zd2911Utils.PowerTimeCount * Zd2911Utils.PowerTimeLength
```

```
                                + cbo_DN.SelectedIndex * Zd2911Utils.PowerTimeLength;
        Array.Copy(data, index, timer, 0, timer.Length);// get the corresponding data according to the NO..
        cbo_PowerHour.SelectedIndex = timer[0];//Hour
        cbo_PowerMinute.SelectedIndex = timer[1];//Minute
        chk_Power.Checked = Convert.ToBoolean(timer[2]);//Enable
}
```

14.11. GetModel

Function: GetProperty(DeviceProperty.Model, extraProperty, ref device, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
extraData = Zd2911Utils.DeviceModel;
bool result = deviceConnection.GetProperty(DeviceProperty.Model, extraProperty, ref device, ref extraData);
if (result) {
    string model = (string)extraData;
}
```

14.12. GetNewlySuperLogData

Function: GetProperty(DeviceProperty.ManagementRecords, extraProperty, ref device, ref extraData)

Example

```
List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
List<bool> boolList = new List<bool>();
boolList.Add(true);//New Admin Logs(the new Slog(menu action logs) that never be downloaded before), when this parameter is true, whether the 2nd parameter "removing the new log mark" is true or false, it will force to remove the new log mark.
boolList.Add(chk_NewFlag.Checked);//removing the new log mark
extraProperty = boolList;
extraData = dtList;//the beginning date, when get the new log, need to set this parameter, but the parameter has no effect.
result = deviceConnection.GetProperty(DeviceProperty.ManagementRecords, extraProperty, ref device, ref extraData);
if (result) {
    List<Record> recordList = (List<Record>)extraData;
```

```
}
```

14.13. GetAllGetSuperLogData

Function: GetProperty(DeviceProperty.ManagementRecords, extraProperty, ref device, ref extraData)

Example

```
List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
List<bool> boolList = new List<bool>();
boolList.Add(false);//All Slogs(MENU action logs)
boolList.Add(chk_NewFlag.Checked);//removing new log mark. when false: not remove.
extraProperty = boolList;
extraData = dtList;//beginning date
result = deviceConnection.GetProperty(DeviceProperty.ManagementRecords, extraProperty, ref device, ref extraData);
if (result) {
    List<Record> recordList = (List<Record>)extraData;
}
```

14.14. GetNewlyGetSuperLogCount

Function: GetProperty(DeviceProperty.ManagementRecordsCount, extraProperty, ref device, ref extraData)

Example

```
List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
extraProperty = true;//new log
extraData = dtList;// the beginning date, when get the new log records count, need to set this parameter, but the parameter has no effect.
result = deviceConnection.GetProperty(DeviceProperty.ManagementRecordsCount, extraProperty, ref device, ref extraData);
if(result) {
    int newlyRecordCount = (int)extraData;
}
```

14.15. GetAllGetSuperLogCount

Function: GetProperty(DeviceProperty.ManagementRecordsCount, extraProperty, ref device, ref extraData)

Example

List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
extraProperty = false;//All logs
extraData = dtList;//the beginning date
result = deviceConnection.GetProperty(DeviceProperty.ManagementRecordsCount, extraProperty, ref device, ref extraData);
if(result) {
int allRecordCount = (int)extraData;
}

14.16. GetNewlyGetGeneralLogData

Function: GetProperty(DeviceProperty.AttRecords, extraProperty, ref device, ref extraData)

Example

List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
List<bool> boolList = new List<bool>();
boolList.Add(true);//New attendance Logs(the new attendance logs that never be downloaded before), when this parameter is true, whether the 2nd parameter "removing the new log mark" is true or false, it will force to remove the new log mark.
boolList.Add(chk_NewFlag.Checked);// remove new log mark.
extraProperty = boolList;
extraData = dtList;// the beginning date, when get the new log, need to set this parameter, but the parameter has no effect.
result = deviceConnection.GetProperty(DeviceProperty.AttRecords, extraProperty, ref device, ref extraData);
if (result) {
List<Record> recordList = (List<Record>)extraData;
}

14.17. GetAllGetGeneralLogData

Function: GetProperty(DeviceProperty.AttRecords, extraProperty, ref device, ref extraData)

Example

List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
List<bool> boolList = new List<bool>();
boolList.Add(false);//All logs
boolList.Add(chk_NewFlag.Checked);//removing new log mark
extraProperty = boolList;
extraData = dtList;//the beginning data
result = deviceConnection.GetProperty(DeviceProperty.AttRecords, extraProperty, ref device, ref extraData);
if (result) {
List<Record> recordList = (List<Record>)extraData;
}

14.18. GetNewlyGetGeneralLogCount

Function: GetProperty(DeviceProperty.AttRecordsCount, extraProperty, ref device, ref extraData)

Example

List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
extraProperty = true;//new logs
extraData = dtList;// the beginning date, when get the new log records count, need to set this parameter, but the parameter has no effect.
result = deviceConnection.GetProperty(DeviceProperty.AttRecordsCount, extraProperty, ref device, ref extraData);
if(result) {
int newlyRecordCount = (int)extraData;
}

14.19. GetAllGetGeneralLogCount

Function: GetProperty(DeviceProperty.AttRecordsCount, extraProperty, ref device, ref

extraData)

Example

List<DateTime> dtList = new List<DateTime>();
dtList.Add(dtp_Begin.Value);
dtList.Add(dtp_End.Value);
object extraProperty = new object();
object extraData = new object();
extraProperty = false;//All logs
extraData = dtList;//the beginning date
result = deviceConnection.GetProperty(DeviceProperty.AttRecordsCount, extraProperty, ref device, ref extraData);
if(result) {
int allRecordCount = (int)extraData;
}

14.20. GetUserEnrollInfo

Function: GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref extraData)

Remark: get user enrollment information list (not including the fingerprint data, password and card NO.)

Example

object extraProperty = new object();
object extraData = new object();
extraProperty = (UInt64)0;
result = deviceConnection.GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref extraData);
if (result) {
List<User> userList = (List<User>)extraData;
}

14.21. GetUserEnrollInfoByUserID

Function: GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref extraData)

Example

object extraProperty = new object();
extraProperty = (UInt64)1;//User DIN
object extraData = new object();
bool result = deviceConnection.GetProperty(DeviceProperty.Enrolls, extraProperty, ref device, ref extraData);
if (result) {
User user = (User)extraData;
}

14.22. GetMacAddress

Function: GetProperty(DeviceProperty.MacAddress, extraProperty, ref device, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
bool result = deviceConnection.GetProperty(DeviceProperty.MacAddress, extraProperty, ref device, ref extraData);
if (result) {
    byte[] bytes = (byte[])extraData;//bytes.Length = 6
}
```

14.23. GetDeviceStatus

Function: GetProperty(DeviceProperty.Status, extraProperty, ref device, ref extraData)

Example

```
object extraProperty = new object();
object extraData = new object();
bool result = deviceConnection.GetProperty(DeviceProperty.Status, extraProperty, ref device, ref extraData);
if (result) {
    UInt32[] count = (UInt32[])extraData;
    //User Count: count[0]
    //Admin Count: count[1]
    //Fingerprint Count: count[2]
    //Card Count: count[3]
    //Password Count: count[4]
    //Newly SLog Count: count[5]
    //Newly GLog Count: count[6]
    //All SLog Count: count[7]
    //All GLog Count: count[8]
}
```

14.24. GetAttendTimeZone

Function: GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData)

Remark: get 3 groups of data at one time (the length of the byte array is 3*14), each set of data is an array of 14 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AttendanceCommand.TimeZone;
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 3 * 14
        byte[] timerZone = new byte[14];
        Array.Copy(data, cbo_AttNo.SelectedIndex * timerZone.Length, timerZone, 0, timerZone.Length);
        cbo_AttBeginHour.SelectedIndex = timerZone[2];//Start Time: Hour
        cbo_AttBeginMinute.SelectedIndex = timerZone[3];//Start Time: Minute
        cbo_AttEndHour.SelectedIndex = timerZone[4];//End Time: Hour
        cbo_AttEndMinute.SelectedIndex = timerZone[5];//End Time: Minute
        nud_AttTimeBeforeOne.Value = timerZone[6];//Time 1 Left
        nud_AttTimeAfterOne.Value = timerZone[7];//Time 1 Right
        nud_AttTimeBeforeTwo.Value = timerZone[8];//Time 2 Left
        nud_AttTimeAfterTwo.Value = timerZone[9];//Time 2 Right
        chk_AttTime1Enable.Checked = Convert.ToBoolean(timerZone[10]);//Time 1 Enable
        chk_AttTime2Enable.Checked = Convert.ToBoolean(timerZone[11]);//Time 2 Enable
        chk_TimeZoneUse.Checked = Convert.ToBoolean(timerZone[12]);//Time Zone Enable
}
```

14.25. GetHoliday

Function: GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData)

Remark: get 40 groups of data at one time (the length of the byte array is 40*10), each set of data is an array of 10 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AttendanceCommand.Holiday;
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 40 * 10
        byte[] holiday = new byte[10];
        // get the corresponding position data according to the NO.
        Array.Copy(data, cbo_HolidayNo.SelectedIndex * holiday.Length, holiday, 0, holiday.Length);
```

```
        dtp_HolidayBeginDate.Value = new DateTime(holiday[2] + 2000, holiday[3], holiday[4]);//Start Date
        dtp_HolidayEndDate.Value = new DateTime(holiday[5] + 2000, holiday[6], holiday[7]);//End Date
}
```

14.26. GetValidAtTime

Function: GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData)

Reamrk: get 24 groups of data at one time (the length of the byte array is 24*6), each set of data is an array of 6 bytes in size.

Example

```
object extraProperty = new object();
object extraData = new object();
extraProperty = AttendanceCommand.LogTime;
extraData = 1;//Attendance Type
result = deviceConnection.GetProperty(DeviceProperty.Attendance, extraProperty, ref device, ref extraData);
if (result) {
    byte[] data = (byte[])extraData;//data.Length = 24 * 6
        byte[] logTime = new byte[6];
        Array.Copy(data, cbo_AttDN.SelectedIndex * logTime.Length, logTime, 0, logTime.Length);// get the corresponding position data according to the NO.
        cbo_BeginHour.SelectedIndex = logTime[2];//Start Time: Hour
        cbo_BeginMinute.SelectedIndex = logTime[3];//Start Time: Minute
        cbo_EndHour.SelectedIndex = logTime[4];//End Time: Hour
        cbo_EndMinute.SelectedIndex = logTime[5];//End Time: Minute
}
```

Chapter 15 Parameter List in Device

Description：settable and obtainable system parameter in Device

SN	Parameter Name	Parameter Index	Remark
1.	Admin Count	0	
2.	Language Format	1	
3.	ID Length	2	
4.	Volume Size	3	
5.	Auto Off Time	4	
6.	Auto Power On	5	
7.	Verify Mode	6	
8.	Auto Learning	7	
9.	Auto Return Time	8	
10.	Standby Time	9	
11.	Enable Alarm In Standby	10	
12.	Card ID Type	11	
13.	Auto Restart	12	
14.	Enable Shutdown	13	
15.	Enable Relay Alarm	14	
16.	Fire Alarm	15	
17.	One To One Security Level	16	
18.	One To N Security Level	17	
19.	SLog Warning Count	18	
20.	GLog Warning Count	19	
21.	Reverify Time	20	
22.	Device ID	21	
23.	Baudrate	22	
24.	User Real Time Log	23	
25.	UDP Port	24	
26.	Device Password	25	
27.	IP Address	26	
28.	Sub Net Address	27	
29.	Default Gate	28	
30.	Server IP Address	29	
31.	Server UDP Port	30	
32.	RS485 Use	31	
33.	Lock Delay Time	32	Belong to the access control settings option
34.	Wiegand Mode	33	Belong to the access control settings option

35.	Check Door State	34	Belong to the access control settings option
36.	Menace Open Door	35	Belong to the access control settings option
37.	Menace Alarm	36	Belong to the access control settings option