Programmer's Guide for Web Solutions

1. Environment Setup

See the procedure running in the **SetupServer.bat** and **SetupClient.bat**. These batch files do the environment setup for the client-server applications.

Server Side

1. Install the IIS(Internet Information Services) 4.0 or above.
2. The server side needs the following files:

WebClient.asp	the main form of the demo
WebEnroll.asp	the form to save the user info to the database.
WebVerify.asp	the form to verify the template of the known user with the user's enrolled template in the database. (1 to 1 matching)
WebIdentify.asp	the form to identify the template of the unknown user with the enrolled templates in the database (1 to N matching). It is suggested the database be not more that 100 templates.
WebSvrMatch.dll	the Dll to do the matching process.
WebFileIO.dll	the Dll to do the database access. The system integrator can replace the dll by saving the templates directly to the actual database such as Access、SQL server、Oracle and etc. All the data are saved in "C:\TestLog".
WIS_Ext.dll	an auxialiary dll to use in WebFileIO.dll.

3. Copy *.asp to C:\InetPub\wwwroot (or another entry point of the server).
4. Copy *.dll to the system32 or any specified folder.
5. The server needs a fixed IP, let's say, 192.168.1.1.
6. Run regsvr32 /s c:\windows\system32\WebSvrMatch.dll to register the Dll, where c:\windows\system32 indicates any location of the dll.
7. Run regsvr32 /s c:\windows\system32\WebFileIO.dll to register the Dll, where c:\windows\system32 indicates any location of the dll.
8. The WebFileIO.dll need the WIS_Ext.dll in the same location.

Client Side

1. The client side needs the following files :

Driver	
WISCMS12.inf **WISCMS12.sys**	Driver for the CMOS reader
API	
WIS_API.ocx	The fingerprint object to run on the browser.
WIS_API.dll	The APIs for use of the OCX.
WISCMS12.dll	The low-level API used by the OCX for CMOS.

2. Copy *.ocx、*.exe and *.dll to the system32 or any specific folder.
3. Run regsvr32 C:\windows\system32\WIS_API.OCX to register the ocx, where C:\windows\system32 indicates any location of the ocx.
4. Run the internet explorer. Select Tools\”Internet Option” and set the safety level to the lowest.

2. Function List

🕘 WIS_API.OCX

All the functions in **WIS_API.OCX** have their corresponding functions in WIS_API.dll. The only difference is that the binary data (BYTE *) in the functions of the OCX or Dll running on the browser are always in ***Variant*** type. See detailed description of each function in the "ProgrammerGuide.pdf".

These functions are listed below:

short **WISInitDriver**(short device);
void **WISTerminateDriver**();
short **WISTestDevice**();
short **WISCheckNoFinger**();
short **WISInitCapture**();
short **WISEndCapture**();
long **WISCapture**(short count);
short **WISCreateTemplate**(VARIANT FAR* rRawTemplate);
short **WISSetEnrollMode**(short Mode);
short **WISEnroll**(VARIANT FAR* rEnrlTemplate);
short **WISReleaseEnroll**();
short **WISVerifyTemplate**(VARIANT FAR& RawTemplate, VARIANT FAR& EnrlTemplate, short Security, long FAR* rScore);
short **WISDisplayImage**(short nStartX, short nStartY, short nDestWidth, short nDestHeight);
BOOL **WISSetParameter**(short Brightness, short Contrast, short Gamma);

There are new functions specially designed for interfacing the TuneImage() function of the web applications. The programmer can also design their own user interface and set the parameters of the image through the WISSetParameter().

BOOL **WISTuneImageInit** (short Brightness, short Contrast, short Gamma) : To set the value of the parameters thorough a displayed dialog.
short **WISGetTunedBrightness**() : To get the value of the current brightness
short **WISGetTunedContrast**() : To get the value of the current contrast.
short **WISGetTunedGamma**() : To get the value of the current gamma.

🕘 WebSvrMatch.dll

All the parameters of the functions running on the server for use of the browser are always in Variant type.

WISServerVerifyTemplate

Synopsis

WISServerVerifyTemplate(VARIANT rawTemplate, VARIANT enrlTemplate, VARIANT securityLevel, VARIANT *rScore, VARIANT *rResult);

Parameter

rawTemplate	The fingerprint code generated through **WISCreateTemplate().**
enrlTemplate	The final fingerprint template generated through **WISEnroll()**.
securityLevel	A parameter to set the threshold that determines where the verification can be passed. See “ProgrammerGuide.pdf”.
rScore	The similarity of two fingerprints to be compared, ranged from 0 ~100. A higher score means a higher similarity.
rResult	The return value. 0 indicates successful verification, otherwise fails.

🕘 WebFileIO.dll

This dll is just for simulating the database process. It provides the functions of saving, loading the enrolled data of the uses. All the data are saved to a predefined directory **“C:\TestLog”**. The programmers may update the functions of the dll by using their own database, such as Access、SQL Server and etc to insert or select the information of the users.

WriteUserData

Synopsis

WriteUserData(VARIANT uid, VARIANT fingerId, VARIANT quality,
VARIANT enrlTemplate);

Parameter

Uid	The id or name of the user to be saved**.**
fingerId	The id of the finger to be saved. 1 ~ 5 corresponds to Right Thumb to Right Little respectively. 6 ~ 10 corresponds to Left Thumb to Left Little respectively.
Quality	The enrolled quality of the finger, must be QUALITY_A ~ QUALITY_D.
enrlTemplate	The enrolled template.

ReadUserData

Synopsis

ReadUserData(VARIANT uid, VARIANT fingerId, VARIANT* rQuality,
VARIANT* rEnrlTemplate, VARIANT* rResult);

Parameter

uid	The id or name of the user to be loaded**.**
fingerId	The id of the finger to be loaded.
rQuality	The loaded enrolled quality of the finger.
rEnrlTemplate	The loaded enrolled template of the user with specified finger Id.
RResult	The return value. ***0*** indicates success. ***-3*** indicates user is not found. ***-1*** indicates finger Id is not between 1 ~10. ***-2*** indicates the id/name is blank.

IsUserExisted

Synopsis

IsUserExisted(VARIANT uid, VARIANT fingerId, VARIANT *rExist);

Parameter

uid	The id or name of the user to be loaded**.**
fingerId	The id of the finger to be loaded.
rExist	The return value. ***1*** indicates existing of the user with specified finger Id. ***0*** indicates user is not found. ***-1*** indicates finger Id is not between 1 ~10. ***-2*** indicates the id/name is blank.

GetUserCount

Synopsis

GetUserCount(VARIANT* rCount);

Parameter

rCount	The return value. Indicates the number of users in the database.

GetUserList

Synopsis

**GetUserList(VARIANT inFlag, VARIANT* rUid, VARIANT* rFingerId,
VARIANT* rQuality, VARIANT* rEnrlTemplate, VARIANT* rResult);**

Description

This function is used for the identification process. The function will automatically move to the next record for subsequent call. And please free all the resource by setting the inFlag to 1 while the loading is terminated.

Parameter

InFlag	***0*** indicates to continue loading the user's info. ***1*** indicates to terminate loading the data and free the resource.
rUid	The id or name of the user currently loaded**.**
rFingerId	The id of the finger currently loaded.
rQuality	The loaded enrolled quality of the finger.
rEnrlTemplate	The loaded enrolled template of the user with specified finger Id.
rResult	The return value. ***0*** indicates success. ***1*** indicates success in freeing the resource. ***-3*** indicates the end of the records. ***-2*** indicates specified user is not found. ***-1*** indicates database is not found.

DeleteUserData

Synopsis
DeleteUserData (VARIANT uid, VARIANT fingerId, VARIANT* rResult);

Parameter

uid	The id or name of the user to be deleted**.**
fingerId	The id of the finger to be deleted.
rResult	The return value. ***0*** indicates success. ***-3*** indicates user is not found. ***-1*** indicates finger Id is not between 1 ~10. ***-2*** indicates the id/name is blank.

◷ Web_Ext.dll

The auxiliary Dll is used to convert the binary data to variant type or vise versa. The data of variant type is need in the use of the browser. In addition, to save the binary template to the database such as Access, the variant type is also required.

WIS_BinaryToVariant

Synopsis

void WINAPI WIS_BinaryToVariant(unsigned char *EnrlTemplate, VARIANT *enrlTemplate, int Size);

Parameter

EnrlTemplate	The input binary template.
enrlTemplate	The output template of Variant type.
Size	The size of the template, should be TEMPLATE_SIZE in this case.

WIS_VariantToBinary

Synopsis

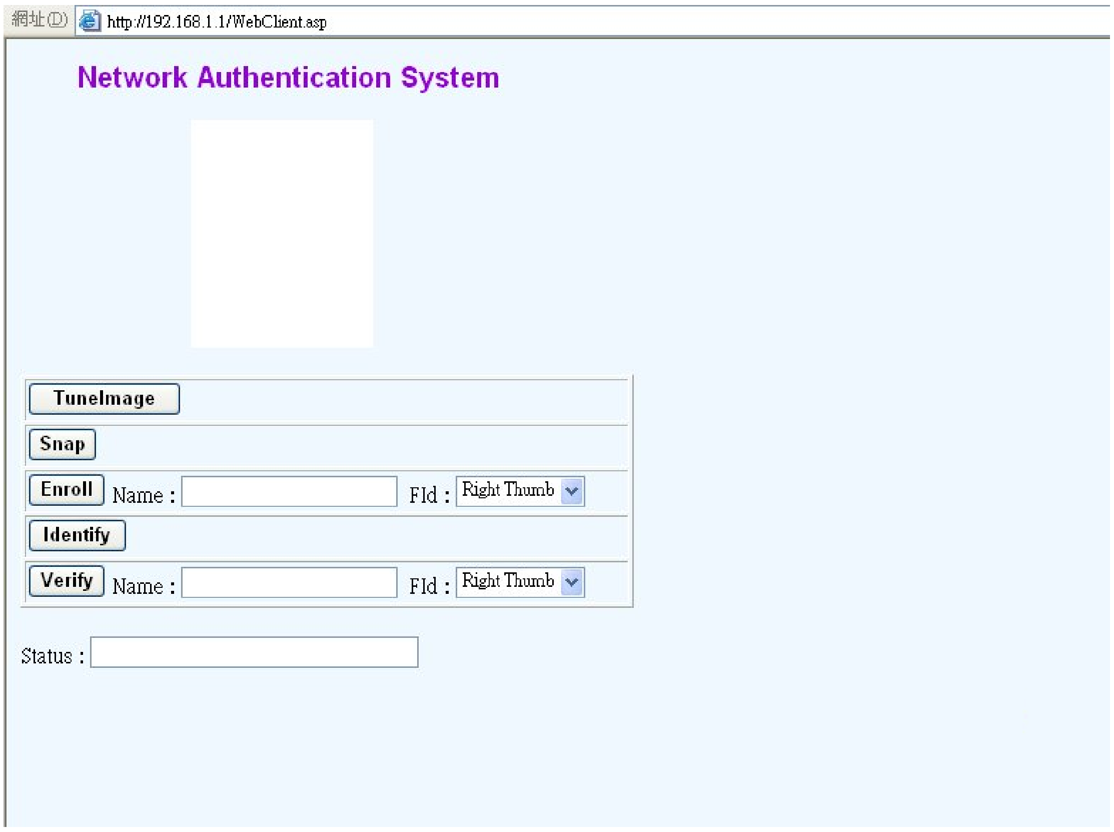
void WINAPI WIS_VariantToBinary (VARIANT *enrlTemplate, unsigned char *EnrlTemplate, int Size);

Parameter

enrlTemplate	The input template of Variant type.
EnrlTemplate	The output binary template.
Size	The size of the template, should be TEMPLATE_SIZE in this case.

3. Running the program

1. Run the internet explorer.
2. Run the demo by typing 192.168.1.1\WebClient.asp, where 192.168.1.1 refer to the fixed IP of the server.
3. The screen below will be shown on the browser.



A. Click “**TuneImage**” to tune the brightness and contrast of the device.
B. Click “**Snap**” to test snapping a fingerprint.
C. Input a name and select a finger and Click “**Enroll**” to enroll a finger. Once succeeded, the user’s info will be sent to the server. All the subsequent process will be done in the WebEnroll.asp.
D. Click “**Identify**” to do the identification process. The template will be sent to the server and all the identification process will be done in the WebIdentify.asp.
E. Input a name and select a finger and Click “**Verify**” to verify a finger. The user’s info will be sent to the server. All the subsequent process will be done in the WebVerify.asp.
F. The status will be shown in the status window.
G. The programmer can change their screen design and flow here.