



DLL instructions document for customized development V1.2

1. EncodeMaterial.dll API Instructions (also can refer to TestEncodeMaterial DLLsource code)

(encode material and convert old project file into new one DLL AIP function) :

```
extern "C" __declspec(dllexport) void __stdcall StartupEncodeMaterial();
extern "C" __declspec(dllexport) void __stdcall ShutDownEncodeMaterial();

extern "C" __declspec(dllexport) int __stdcall LxdToXml(char *cLxdName, char *cXmlName);
extern "C" __declspec(dllexport) int __stdcall ComplierTemperatureHumidity(char *cFileName,
char *cFontName,
int iFontSize,
int iFontColor,
int iBackColor,
char *cTemperateDescription,
char *cHumidityDescription);

extern "C" __declspec(dllexport) int __stdcall ComplierDigitalClock(char *cFileName,
char *cFontName,
int iFontSize,
int iFontColor,
int iBackColor,
char *cTextDescription);

extern "C" __declspec(dllexport) int __stdcall ComplierAnalogClock(char *cFileName, //material file path name
int iBackColor, //background color
int uiAnalobClockXingZhuang, //clock shape, 0 roundness, 1 rectangle
int uiHourXingZhuang, //hour pointer shape, //0 roundness, 1 rectangle
int uiHourPointSize, //hour pointer size
int crHourPoint, //hour pointer color
int uiMinuteXingZhuang, //minute pointer shape, // 0 roundness, 1 rectangle
int uiMinutePointSize, //minute pointer size
int crMinutePoint, // minute pointer color
int crHourGuidelines, //hour hand color
int crMinuteGuidelines, //minute color
int crSecondGuidelines, //second color
char *cAddress, //clock region content, for example: Beijing
char *cAddressFontName, //clock region font
int iAddressFontSize, //clock region font size
int iAddressFontColor, //clock region font color
char *cDateFontName, //date font
int iDateFontSize, //date font size
```



```
extern "C" __declspec(dllexport) int __stdcall  ComplierTimer(char *cFileName,
```

```
int iDateFontColor,//date font color
char *cWeekFontName,//week font
int iWeekFontSize,//week font size
int iWeekFontColor,//week font color
int iWndWidth,//width of dial plate
int iWndHeight//height of dial plate
);
```

```
char *cFontName,
int iFontSize,
int iFontColor,
int iBackColor,
char *cTextDescription
);
```

2. XiXunLedProgram.dll API Instructions (operate xixun xml project file DLL API function):

```
// The following ifdef block is the standard way of creating macros which make exporting
// from a DLL simpler. All files within this DLL are compiled with the XIXUNLEDPROGRAM_EXPORTS
// symbol defined on the command line. This symbol should not be defined on any project
// that uses this DLL. This way any other project whose source files include this file see
// XIXUNLEDPROGRAM_API functions as being imported from a DLL, whereas this DLL sees symbols
// defined with this macro as being exported.
//xixun LED program XML format's interface
//date: 2012-11-0
//update date: 2012-12-03
//update content:
//1,all of ItemName belongs to the only one windowName and PageName  <ITEM_INFO structure > saved in list_item linked list.
//2, all of windowName belongs to the only one pageName <WINDOW_INFO structure > saved in list_window linked list.
//3, all of pageName must be single one and saved in list_page linked list.
//4.when CreateProgram need to clear up all of three linked lists.
//5.when LoadProgram put pageName, windowName and itemName of xml into these three linked list.

//update date:2012-12-11
//update content:
//1, add new functions of CI_AddBackgroundMusic and CI_RemoveBackgroundMusic, to adding background music for webpage.
//2, enable nWindowAmount property in PageAttribute, after add new window and delete it will update this property values.
```



//3, abandon wsBackgroundMusicPath in PageAttribute <save and show one group of property in xsd >

//update date: 2012-12-13

//update content:

//add nFontWeight and nTextColor property in TextFileAttribute

//update date: 2012-12-18

//update content:

//1, add dial shape property for nAnalogClockShape in AnalogClockWindow

//2, add week mark for bShowWeek in AnalogClockWindow

//3, add timer type for nTimerType in PeroidWindow

//4, add week property for DateTime in PeroidWindow

//update date: 2012-12-24

//update content:

//1, add variant name property (wsPageVariantName) for PageAttribute

//2, add only show maximum unit (bMaxUnitOnly) , multi window view (bMultiView) and

// show time unit (bShowTimeUnit) properties for PeriodWindow

//3, add ItemName property for BackgroundMusic in PageAttribute

// interface initialing and releasing -----//

void CI_InitDLL();

void CI_UnInitDLL();

//---- Proram -----//

//upload an existed program via xml file path

BOOL CI_LoadProgram(wchar_t* xmlFile);

void CI_SaveProgram(wchar_t* xmlFile);

//build a new program and its attributes (program page numbers will update automatically according to the pages)

void CI_CreateProgram(struct ProgramAttribute* pAttr);

//get and set program attributes

BOOL CI_GetProgramAttribute(struct ProgramAttribute* pAttr);

BOOL CI_SetProgramAttribute(struct ProgramAttribute* pAttr);

//-----//

//---- Pages + Page -----//

//build a new program page and its attributes, return its name (function will create windows node for each program automatically)

wchar_t* CI_CreatePage(struct PageAttribute* pAttr);



```
BOOL CI_RemovePage(wchar_t* wsPageName);
//get and set program page attributes
BOOL CI_GetPageAttribute(wchar_t* wsPageName, PageAttribute* pAttr);
BOOL CI_SetPageAttribute(wchar_t* wsPageName, PageAttribute* pAttr);

//build a new global program page and its attributes, return its name ( this function can be called only one time )
wchar_t* CI_CreateGlobalPage(struct PageAttribute* pAttr);
//remove global program page can call Removepage ( )
//get and set attributes call CI_GetPageAttribute和CI_SetPageAttribute
//get and set program page constraint
BOOL CI_AddConstraint(wchar_t* wsPageName, wchar_t* wsItemName,
    struct DateConstraint* ds, struct TimeConstraint* ts, struct WeekConstraint* ws);
//remove one program page display constraint
BOOL CI_RemoveConstraint(wchar_t* wsPageName, wchar_t* wsItemName);
//-----//

//---- Windows + FileWindow -----//
//build a new window, need input two parameters: program page name and file window attributes
wchar_t* CI_CreateFileWindow(wchar_t* wsPageName, struct FileWindowAttribute* pAttr);
//get and set file window attributes
BOOL CI_GetFileWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, struct FileWindowAttribute* pAttr);
BOOL CI_SetFileWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, struct FileWindowAttribute* pAttr);
//remove file window
BOOL CI_RemoveWindow(wchar_t* wsPageName, wchar_t* wsWindowName);
//add material to file window
BOOL CI_AddImageToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct PictureAttribute* pAttr);
BOOL CI_AddTextToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct TextFileAttribute* pAttr);
BOOL CI_AddVideoToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct VideoAttribute* pAttr);
BOOL CI_AddGIFToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct GIFAttribute* pAttr);
BOOL CI_AddRTFTToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct RTFAttribute* pAttr);
BOOL CI_AddFlashToFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, struct FlashAttribute* pAttr);
BOOL CI_RemoveItemFromFileWindow(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsItemName);
//-----//

//---- windows + SingLineTextWindow -----//
//build a single line text window and return its name
wchar_t* CI_CreateSinglineTextWindow(wchar_t* wsPageName,
    struct TextWindowAttribute* pWindow,
    struct SingLineTextAttribute* pAttr,
    struct TextAttribute* pText);
```



```
//remove a SinglineTextWindow, same as CI_RemoveWindow()
//XIXUNLEDPROGRAM_API bool CI_RemoveWindow(wchar_t* wsWindowName);
//get and set TextWindow attributes and text
BOOL CI_GetSinglineTextAttribute(wchar_t* wsPageName,
                                wchar_t* wsWindowName,
                                struct TextWindowAttribute* pWindow,
                                struct SingLineTextAttribute* pAttr,
                                struct TextAttribute* pText=0);

BOOL CI_SetSinglineTextAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                struct TextWindowAttribute* pWindow,
                                struct SingLineTextAttribute* pAttr,
                                struct TextAttribute* pText=0);

BOOL CI_GetText(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsItemName, struct TextAttribute* pText);
BOOL CI_SetText(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsItemName, struct TextAttribute* pText);
//-----//

BOOL CI_GetTextWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, struct TextWindowAttribute* pAttr);
BOOL CI_SetTextWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, struct TextWindowAttribute* pAttr);
//---- windows + MultiLineTextwindow -----//
//build a multi-line text window and return its name
wchar_t* CI_CreateMultilineTextWindow(wchar_t* wsPageName,
                                       struct TextWindowAttribute* pWindow);

//remove a MultilineTextWindow, same like CI_RemoveWindow()
//add a new text into multi-line text window
BOOL CI_AddMultiTextToWindow(wchar_t* wsPageName, wchar_t* wsWindowName,
                             struct MultiLineTextAttribute* pAttr, struct TextAttribute* pText);

//get and set
BOOL CI_GetMultilineTextAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                  wchar_t* wsItemName, struct TextWindowAttribute* pWindow,
                                  struct MultiLineTextAttribute* pAttr, struct TextAttribute* pText=0);

BOOL CI_SetMultilineTextAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsItemName,
                                  struct MultiLineTextAttribute* pAttr, struct TextAttribute* pText);

//get and set Text, same like CI_GetText() and CI_SetText()
//remove one line text
BOOL CI_RemoveText(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsItemName);
//-----//

//---- windows + TableWindow -----//
//build a table window
wchar_t* CI_CreateTableWindow(wchar_t* wsPageName, struct TableWindowAttribute* pAttr);
```



```
//remove, same like CI_RemoveWindow ( )
//get and set TableWindow attributes
BOOL CI_GetTableWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                struct TableWindowAttribute* pAttr);
BOOL CI_SetTableWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                struct TableWindowAttribute* pAttr);

//add a new table
BOOL CI_AddTable(wchar_t* wsPageName, wchar_t* wsTableWindowName, struct TableAttribute* pAttr,
                 struct TextAttribute* pTitle=0);
BOOL CI_RemoveTable(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsTableName);
BOOL CI_GetTableAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsTableName, struct TableAttribute* pAttr);
BOOL CI_SetTableAttribute(wchar_t* wsPageName, wchar_t* wsWindowName, wchar_t* wsTableName, struct TableAttribute* pAttr);
//-----//

//---- windows + PeroidWindow -----//
//build a period window
wchar_t* CI_CreatePeroidWindow(wchar_t* wsPageName,
                               struct PeroidWindowAttribute* pAttr,
                               struct DATE_TIME * pDt,
                               struct TextAttribute* pTitle=0);

//remove, same like CI_RemoveWindow ( )
//get and set PeroidWindow attributes
BOOL CI_GetPeroidWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                 struct PeroidWindowAttribute* pAttr);
BOOL CI_SetPeroidWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                 struct PeroidWindowAttribute* pAttr);

//get and set title, same like CI_GetText() and CI_SetText()
//-----//

//---- windows + ClockWindow -----//
//build a digital clock window
wchar_t* CI_CreateDigitalClockWindow(wchar_t* wsPageName,
                                     struct DigitalClockWindowAttribute* pAttr, struct TextAttribute* pTitle=0);
//remove, same like CI_RemoveWindow ( )

//get and set digital clock window attributes
BOOL CI_GetDigitalClockWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                       struct DigitalClockWindowAttribute* pAttr);
BOOL CI_SetDigitalClockWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
                                       struct DigitalClockWindowAttribute* pAttr);
```



```
//build an analog clock window
wchar_t* CI_CreateAnalogClockWindow(wchar_t* wsPageName,
    struct AnalogClockWindowAttribute* pAttr, struct TextAttribute* pTitle=0,
    struct TextAttribute* pDateTitle=0, struct TextAttribute* pWeekTitle=0);
//remove, same like CI_RemoveWindow ( )

//get and set analog clock window attributes
BOOL CI_GetAnalogClockWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
    struct AnalogClockWindowAttribute* pAttr);
BOOL CI_SetAnalogClockWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
    struct AnalogClockWindowAttribute* pAttr);

//get and set title, same like CI_GetText() and CI_SetText()
//-----//

//---- windows + TempHumiWindow -----//
//build a temperature and humidity window
wchar_t* CI_CreateTempHumiWindow(wchar_t* wsPageName,
    struct TemperatureHumidityWindowAttribute* pAttr);
//remove, same like CI_RemoveWindow()
//get and set TempHumiWindow attributes
BOOL CI_GetTempHumiWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
    struct TemperatureHumidityWindowAttribute* pAttr);
BOOL CI_SetTempHumiWindowAttribute(wchar_t* wsPageName, wchar_t* wsWindowName,
    struct TemperatureHumidityWindowAttribute* pAttr);
//-----//

//initializing attributes -----//
void CI_InitTextFile(struct TextFileAttribute* pAttr);
void CI_InitText(struct TextAttribute* pAttr);
void CI_InitPicture(struct PictureAttribute* pAttr);
void CI_InitVideo(struct VideoAttribute* pAttr);
void CI_InitGIF(struct GIFAttribute* pAttr);
void CI_InitRTF(struct RTFAttribute* pAttr);
void CI_InitFlash(struct FlashAttribute* pAttr);
void CI_InitProgram(struct ProgramAttribute* pAttr);
void CI_InitPage(struct PageAttribute* pAttr);
void CI_InitFileWindow(struct FileWindowAttribute* pAttr);
void CI_InitSinglineText(struct SingLineTextAttribute* pAttr);
```



```
void CI_InitMultilineText(struct MultiLineTextAttribute* pAttr);
void CI_InitTableWindow(struct TableWindowAttribute* pAttr);
void CI_InitTable(struct TableAttribute* pAttr);
void CI_InitPeroidWindow(struct PeroidWindowAttribute* pAttr);
void CI_InitDigitalClockWindow(struct DigitalClockWindowAttribute* pAttr);
void CI_InitAnalogClockWindow(struct AnalogClockWindowAttribute* pAttr);
void CI_InitTempHumiWindow(struct TemperatureHumidityWindowAttribute* pAttr);
void CI_InitTextWindow(struct TextWindowAttribute* pAttr);
//-----//
BOOL CI_ValidateProgram(wchar_t* pErrMsg, int nMsgLength);
void CI_SetLastError(int nError);
int CI_GetLastError();
wchar_t* CI_GetVersion();

//-----//
-----//
Structure definition for reference as following:
#define ATT_BUFF_SIZE 1024
// ----- basis element-----//
//text files type --12--
struct TextFileAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE];//text file custom name
    wchar_t wsTextFilePath[ATT_BUFF_SIZE];//file path name of text file
    wchar_t wsTextFont[ATT_BUFF_SIZE];//font name
    wchar_t wsBackground[ATT_BUFF_SIZE];//background picture file
    int nBackColor;//background color
    int nTextFontStyle;//font style (bold 0x4,italics 0x2,underline 0x1)
    int nSpecialEffects;//special effects type
    int nSpecialEffectSpeed;//special effects speed
    int nSpecialEffectDuration;//stay time
    int nTextLineSpace;//line space
    int nViewMode;//display mode, reserved
    int nVerPosType;//0 vertical center ,1vertical top,2vertical bottom
    //int nPlayLoopCount;
};

//text type -- 6 --
struct TextAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE];// text custom name
    wchar_t wsText[ATT_BUFF_SIZE];//text content
```




```
wchar_t wsFont[ATT_BUFF_SIZE]; //font
int nFontWeight; //font size
int nTextColor; //font color
int nTextStyle; //font style (bold 0x4, italics 0x2, underline 0x1)
};

//image type --- 9 ---
struct PictureAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE]; //unique identification name
    wchar_t wsFilePath[ATT_BUFF_SIZE]; //file path name
    int nViewMode; //display mode 0:center 1:zoom 2:stretch 3:title
    int nBackColor; //background color
    int nSpecialEffects; //special effects
    int nSpecialEffectSpeed; //special effects speed
    int nSpecialEffectDuration; //stay time
    int nFinishSpecialEffects; //clear effects
    int nFinishSpecialEffectSpeed; // clear effects speed
};

//video type -- 6
struct VideoAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE]; //unique identification name
    wchar_t wsFilePath[ATT_BUFF_SIZE]; //file path name
    int nPlayLoopCount; //loop count
    int nBackColor; //background color
    int nVideoWidth; //video width
    int nVideoHeight; //video height
};

//GIF type ---5
struct GIFAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE]; //unique identification name
    wchar_t wsFilePath[ATT_BUFF_SIZE]; //file path name
    int nBackColor; //loop count
    int nSpecialEffects; //display mode, 0: center 1:zoom 2:stretch 3:title
    int nPlayLoopCount; //loop count
};

//RTF type -- 10
struct RTFAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE]; //unique identification name
```



```
wchar_t wsFilePath[ATT_BUFF_SIZE];//file path name
wchar_t wsBackground[ATT_BUFF_SIZE];//background picture
int nBackColor;//background color
int nSpecialEffects;//special effects type
int nSpecialEffectSpeed;//special effects speed
int nSpecialEffectDuration;//stay time
int nLineSpace;//line space
int nVerPosType;//0vertical center,1vertical top,2 vertical bottom
int nDisplayType;//display mode, reserved
};

//Flash type-- 9
struct FlashAttribute{
    wchar_t wsItemName[ATT_BUFF_SIZE];//unique identification name
    wchar_t wsFilePath[ATT_BUFF_SIZE];//file path name
    int nBackColor;//background color
    int nFlashWidth;//width
    int nFlashHeight;//height
    int nFrameRate;//frame rate
    int nStartFrame;//start frame
    int nFinishFrame;//end frame
    int nPlayLoopCount;//loop count
};
// -----//

//program attributes-- 13
struct ProgramAttribute{
    wchar_t wsVersion[ATT_BUFF_SIZE];//version
    int nDirectionX;// x coordinate start point of preview window on PC
    int nDirectionY;// y coordinate start point of preview window on PC
    int nScreenWidth;//program width
    int nScreenHeight;//program height
    int nCommunicationType;//communication type, 0:network,1:serial port,2:U-disk,3:GPRS,4:leddata server center,5:FTP
    wchar_t wsIPAddress[ATT_BUFF_SIZE];//IP address
    wchar_t wsLanguage[ATT_BUFF_SIZE];//language type
    int nPort;// network port or serial port
    int nBaudrate;//baud rate
    int nUSBMode;//U-disk mode,0:update program,1:extend storage
    bool bFullScreen;//full screen or not ,reserved
    bool bTextSmooth;//text smooth or not, reserved
```



```
};

//Page attributes -- 7
struct PageAttribute{
    wchar_t wsPageName[ATT_BUFF_SIZE];//program page unique identification
    int nWindowAmount;//window numbers
    wchar_t wsBackgroundPath[ATT_BUFF_SIZE];//background picture
    int nBackgroundColor;//background color
    wchar_t wsBackgroundMusicPath[ATT_BUFF_SIZE];//background music
    int nPlayDuration;//play time
    bool bWaitFinish;//wait for end, true: program page finished after displaying all of its content for one time ,false: display program page according to play time,
};

//FileWindow attributes -- 8
struct FileWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE];//window unique identification
    wchar_t wsBackgroundPath[ATT_BUFF_SIZE];//background picture
    int nWindowFrame;//frame
    int nWindowFrameColor;//frame color
    int nDirectionX;// x coordinate start point of window
    int nDirectionY;//y coordinate start point of window
    int nWindowWidth;//window width
    int nWindowHeight;//window height
};

//text window attributes --8--
struct TextWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE];//window unique identification
    int nWindowBackColor;//background color
    int nWindowFrame;//frame
    int nWindowFrameColor;//frame color
    int nDirectionX;// x coordinate start point of window
    int nDirectionY;// y coordinate start point of window
    int nWindowWidth;// window width
    int nWindowHeight;// window height
};

//single line text -- 10
struct SingLineTextAttribute{
    int nSpecialEffects;//special effects
```



```
int nSpecialEffectSpeed;// special effects speed
int nSpecialEffectDuration;//stay time
wchar_t wsBackgroundPath[ATT_BUFF_SIZE];//background picture
bool bPictureUseasIcon;//true: background picture used for logo,false:background picture used for background
int nVerticalOffset;//vertical offset
int nTextSpace;//text space
int nTextRotaion;//0:no rotate ,1:90 rotate ,2:180 rotate,3:270 rotate
int nTextStyle;//font style (bold 0x4,italics 0x2,underline 0x1)
int nTextStyleColor;//font color
};

//multi-line text -- 9
struct MultiLineTextAttribute{
    wchar_t wsTextName[ATT_BUFF_SIZE];//text unique identification
    int nTextHorizonAligning;//0:left align,1:right align ,2:center align ,3justify align
    int nTextVerticalAligning;//0vertical center,1vertical top,2vertical bottom
    int nSpecialEffects;//special effects
    int nSpecialEffectDuration;//stay time
    int nSpecialEffectSpeed;// special effects speed
    int nBackgroundColor;//background color
    wchar_t wsBackgroundPicturePath[ATT_BUFF_SIZE];//background picture
    int nTextSpace;//text space
    int nLineSpace;//line space
};

//table window attributes  -- 8
struct TableWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE];//window unique identification
    int nWindowFrame;//frame
    int nWindowFrameColor;//frame color
    int nWindowBackColor;//background color
    int nDirectionX;// x coordinate start point of window
    int nDirectionY;// y coordinate start point of window
    int nWindowWidth;//window width
    int nWindowHeight;//window height
};

//table attributes -- 10
struct TableAttribute{
    wchar_t wsTableName[ATT_BUFF_SIZE];//table unique identification
```



```
wchar_t wsFilePath[ATT_BUFF_SIZE]; //file path name
int nRowAmount; //row amount
int nColumnAmount; //column amount
int nFixedRowAmount; //fixed row amount
int nFixedColumnAmount; //fixed column amount
int nGridLineColor; //table color
int nSpecialEffects; // special effects
int nSpecialEffectSpeed; // special effects speed
int nSpecialEffectDuration; //stay time
};

struct DATE_TIME{
    int nYear; //year
    int nMonth; //month
    int nDay; //day
    int nHour; //hour
    int nMinute; //minute
    int nSecond; //second
};

//timer window attributes-- 16 ---
struct PeroidWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE]; // window unique identification
    wchar_t wsBackgroundPath[ATT_BUFF_SIZE]; //background picture
    int nWindowFrame; //frame
    int nWindowFrameColor; //frame color
    int nWindowBackColor; //background color
    int nDirectionX; // x coordinate start point of window
    int nDirectionY; // y coordinate start point of window
    int nWindowWidth; //window width
    int nWindowHeight; //window height
    struct DATE_TIME definiteTime; //date time structure
    bool bShowDays; //show day or not
    bool bShowHours; //show hour or not
    bool bShowMinutes; //show minutes or not
    bool bShowSeconds; //show second or not
    bool bShowMultiLine; //true: show in multi-line false: show in single line
    bool bUnsignedNumber; //show negative number or not
};

//digital clock window attributes --29--
struct DigitalClockWindowAttribute{
```



```
wchar_t wsWindowName[ATT_BUFF_SIZE];// window unique identification
wchar_t wsBackgroundPath[ATT_BUFF_SIZE];// background picture
int nWindowFrame;// rame
int nWindowFrameColor;// frame color
int nWindowBackColor;// background color
int nDirectionX;// x coordinate start point of window
int nDirectionY;// y coordinate start point of window
int nWindowWidth;//window width
int nWindowHeight;//window height
int nShowStyle;//show style,1:[2003-05-30],2:[05-30-2003],3:[30-05-2003],4:[2003/05/30],5:[05/30/2003],6:[30/05/2003],7:[2003 year 5 month 30 day]
int nHourStyle;//0:12-hour ,1:24-hour
int nYearStyle;//0:4 digits year 1:2 digits year
bool bShowYear;//show year or not
bool bShowMonth;//show month or not
bool bShowDay;//show day or not
bool bShowLunarYear;//show Lunar year or not
bool bShowLunarMonth;//show Lunar month or not
bool bShowLunarDay;//show Lunar day or not
bool bShowWeek;//show week or not
bool bShowHalfDay;//show half day or not
bool bShowHour;//show hour or not
bool bShowMinute;//show minute or not
bool bShowSecond;// show second or not
bool bHourrateBefore;//true: time before, false: time lag
int nHourrateDays;//hourrate days
int nHourrateHours;//hourrate hours
int nHourrateMinutes;// hourrate minutes
int nHourrateSeconds;// hourrate seconds
int nMultiLineStyle;//0: show in single line ,1:show in multi-line
};

//analog clock window attributes ---25---
struct AnalogClockWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE];// window unique identification
    wchar_t wsBackgroundPath[ATT_BUFF_SIZE];// background picture
    int nWindowFrame;// frame
    int nWindowFrameColor;// frame color
    int nWindowBackColor;//background color
    int nDirectionX;// x coordinate start point of window
    int nDirectionY;// y coordinate start point of window
    int nWindowWidth;//window width
```



```
int nWindowHeight;//window height
int nHourScaleColor;//hour scale color
int nHourScaleSize;//hour scale size
int nHourScaleShape;//hour scale shape,0: roundness,1:rectangle ,2:number
int nMinuteScaleColor;//minute scale color
int nMinuteScaleSize;// minute scale size
int nMinuteScaleShape;// minute scale shape, 0: roundness,1:rectangle
bool bShowDate;//show date or not
bool bLunarCalendar;//show Lunar calendar or not
int nHourhandColor;//hour hand color
int nMinutehandColor;//minute hand color
int nSecondhandColor;//second hand color
int nHourrateDays;//hourrate days
int nHourrateHours;// hourrate hours
int nHourrateMinutes;// hourrate minutes
int nHourrateSeconds;// hourrate seconds
bool bHourrateBefore;//true: time before ,false: time lag
};

struct TemperatureHumidityWindowAttribute{
    wchar_t wsWindowName[ATT_BUFF_SIZE];// window unique identification
    int nWindowFrame;//frame
    int nWindowFrameColor;//frame color
    int nWindowBackColor;//background color
    int nDirectionX;// x coordinate start point of window
    int nDirectionY;// y coordinate start point of window
    int nWindowWidth;//window width
    int nWindowHeight;//window height
    int nFontWeight;//font size
    int nTextColor;//font color
    int nFontStyle;//font style(bold 0x4,italics 0x2,underline 0x1)
    int nDegreeType;//temperature type,0: centigrade ,1:fahrenheit
    int nTempRegulate;//temperature regulate
    int nHumiRegulate;//humidity regulate
    bool bMultiLine;//0:show in single line ,1:show in multi-line
    bool bShowTemperature;//show temperature or not
    bool bShowHumidity;//show humidity or not
    bool bShowTemperatureUnit;//show temperature unit or not
    bool bShowHumidityUnit;//show humidity unit or not
    wchar_t wsFont[ATT_BUFF_SIZE];//font
    wchar_t wsTempTitle[ATT_BUFF_SIZE];//temperature title
};
```



```
wchar_t wsHumiTitle[ATT_BUFF_SIZE];//humidity title
wchar_t wsHumidityUnit[ATT_BUFF_SIZE];//temperature unit
wchar_t wsTemperatureUnit[ATT_BUFF_SIZE];//humidity unit
wchar_t wsBackgroundPath[ATT_BUFF_SIZE];//background picture path file
};

//page constraint
struct DateConstraint{
    int nStartYear;//start year
    int nStartMonth;// start month
    int nStartDay;// start day
    int nEndYear;//end year
    int nEndMonth;//end month
    int nEndDay;//end day
};

struct TimeConstraint{
    int nStartHour;//start hour
    int nStartMinute;//start minute
    int nStartSecond;//start second
    int nEndHour;//end hour
    int nEndMinute;//end minute
    int nEndSecond;//end second
};

struct WeekConstraint{
    bool bMonday;//Monday
    bool bTuesday;//Tuesday
    bool bWednesday;//Wednesday
    bool bThursday;//Thursday
    bool bFriday;//Friday
    bool bSaturday;//Saturday
    bool bSunday;//Sunday
};

enum COMMUNICATION_TYPE{NETWORK=1, COMPORT, USBMODE};
enum TEXT_STYLE{STANDARD=0, BOLD=0x4, ITALIC=0x2, UNDERLINE=0x1};
enum VIEW_MODE{CENTER, ZOOM, STRETCH, TILING};
enum TEXT_ROTATION{ROTATE_0, ROTATE_90, ROTATE_180, ROTATE_270};
enum TEXT_ALIGNING{ALIGNING_NONE, ALIGNING_LEFT, ALIGNING_CENTER, ALIGNLING_RIGHT, ALIGNLING_UP, ALIGNLING_DOWN};
```




```
enum SHAPE{SHAPE_ROUND, SHAPE_RECT};
enum TIMER_STYLE{COUNT, COUNTDOWN};
enum ERROR_VALUES{ERROR_PROGRAM_EXIST = 1, ERROR_PROGRAM_ATTR, ERROR_IN_PARAM, ERROR_OUT_PARAM,
    ERROR_PAGE_NAME, ERROR_EXIST_GLOBALPAGE, ERROR_WINDOW_NAME,
    ERROR_WINDOW_NAME_NOT_FOUND, ERROR_PAGE_NAME_NOT_FOUND, ERROR_ITEM_NAME,
    ERROR_PAGE_NAME_EXIST, ERROR_WINDOW_NAME_EXIST, ERROR_TABLE_NAME_EXIST,
    ERROR_ITEM_NAME_EXIST, ERROR_ITEM_NAME_NOT_FOUND};
enum _ENUM_ACTION
{
    ENUM_TEXIAO_NO=0,
    ENUM_TEXIAO_RADIO,
    ENUM_TEXIAO_FG_UP,
    ENUM_TEXIAO_FG_DOWN,
    ENUM_TEXIAO_FG_LEFT,
    ENUM_TEXIAO_FG_RIGHT,
    ENUM_TEXIAO_FG_CORNER_RD,
    ENUM_TEXIAO_FG_CORNER_LD,
    ENUM_TEXIAO_FG_CORNER_RU,
    ENUM_TEXIAO_FG_CORNER_LU,
    ENUM_TEXIAO_FG_UD_IN,
    ENUM_TEXIAO_FG_LR_IN,
    ENUM_TEXIAO_FG_UD_OUT,
    ENUM_TEXIAO_FG_LR_OUT,
    ENUM_TEXIAO_FG_SZ_IN,
    ENUM_TEXIAO_FG_SZ_OUT,
    ENUM_TEXIAO_FG_OUT_INT,
    ENUM_TEXIAO_FG_INT_OUT,
    ENUM_TEXIAO_MOVEUP,
    ENUM_TEXIAO_MOVEDOWN,
    ENUM_TEXIAO_MOVELEFT,
    ENUM_TEXIAO_MOVERIGHT,
    ENUM_TEXIAO_MOVERIGHTDOWN,
    ENUM_TEXIAO_MOVELEFTDOWN,
    ENUM_TEXIAO_MOVERIGHTUP,
    ENUM_TEXIAO_MOVELEFTUP,
    STUNT_COVER_SLANT_TOP_LEFT      , // cover top left (slant) ;
    STUNT_COVER_SLANT_TOP_RIGHT     , // cover top right (slant) ;
    STUNT_COVER_SLANT_BOTTOM_LEFT   , // cover bottom left (slant) ;
    STUNT_COVER_SLANT_BOTTOM_RIGHT  , // cover bottom right (slant) ;
    STUNT_BLIND_HORIZONE             , // blind horizontal; each page width is 8 for each window.
```



STUNT_BLIND_VERTICAL	, // blind vertical; Each page width is 8 for each window.
STUNT_MASAIC	, // mosaic, each mosaic size is 8x8 pixels.
STUNT_CIRCLE_OUT	, // circle out
STUNT_CIRCLE_IN	, // circle in
STUNT_TURN_CIRCLE_LEFT_360	, // spin left 360 degree
STUNT_TURN_CIRCLE_RIGHT_360	, // spin right 360 degree
STUNT_TURN_CIRCLE_LEFT_180	, // spin left 180 degree
STUNT_TURN_CIRCLE_RIGHT_180	, // spin right 180 degree
STUNT_TURN_CIRCLE_LEFT_90	, // spin left 90
STUNT_TURN_CIRCLE_RIGHT_90	, // spin right 90
STUNT_SECTOR	, // fan shape
STUNT_ZOOM_UP	, // zoom (up)
STUNT_ZOOM_DOWN	, // zoom (down)
STUNT_ZOOM_LEFT	, // zoom (left)
STUNT_ZOOM_RIGHT	, // zoom (right)
STUNT_ZOOM_CENTROL	, // zoom (center)
STUNT_ZOOM_TOP_LEFT	, // zoom (top left)
STUNT_ZOOM_TOP_RIGHT	, // zoom (top right)
STUNT_ZOOM_BOTTOM_LEFT	, // zoom (bottom left)
STUNT_ZOOM_BOTTOM_RIGHT	, // zoom (bottom right)
STUNT_STUNT_COVER_RHOMBUS_IN	, // cover in (rhombus)
STUNT_STUNT_COVER_RHOMBUS_OUT	, // cover out (rhombus)
STUNT_STUNT_COVER_CIRCLE_IN	, // cover in (circle)
STUNT_STUNT_COVER_CIRCLE_OUT	, // cover out (circle)
STUNT_MERGE_HORIZONE	, // merge horizontal
STUNT_MERGE_VERTICAL	, // merge vertical
STUNT_CHEQUIRE_HORIZONE	, // chessboard horizontal
STUNT_CHEQUIRE_VERTICAL	, // chessboard vertical
STUNT_GLINT_HORIZONE	, // blink horizontal
STUNT_GLINT_VERTICAL	, // blink vertical
STUNT_INTERSECT_HORIZONE	, // cross horizontal
STUNT_INTERSECT_VERTICAL	, // cross vertical
STUNT_SNOW	, // snow
STUNT_FALLOW_DOWN	, // drop down
STUNT_GILNT	, // blink, can define times of blink, PC software default 3 times, 0 stands for blink forever
STUNT_AGGLOMERTION_UP	, // coalescence upward
STUNT_AGGLOMERTION_DOWN	, // coalescence download
STUNT_AGGLOMERTION_LEFT	, //coalescence left
STUNT_AGGLOMERTION_RIGHT	, // coalescence right
STUNT_LASER_UP	, // laser up



```
STUNT_LASER_DOWN           , // laser down
STUNT_LASER_LEFT           , // laser left
STUNT_LASER_RIGHT          , // laser right
STUNT_STRETCH_UP           , // stretch up
STUNT_STRETCH_DOWN         , // stretch down
STUNT_STRETCH_LEFT         , // stretch left
STUNT_STRETCH_RIGHT        , // stretch right
STUNT_CENTROL_MOVE_OUT     , // middle out
STUNT_GRADUAL_CHANGE       , // transition effect, image brightness changes from 0 to 255 according to 16 stepping. Controller without gray
level does not have this option
STUNT_BANMATIAO_HORIZONE   , //zebra horizontal
STUNT_BANMATIAO_VERTICAL   , //zebra vertical
STUNT_MASAIC_SMALL         , // mosaic, each mosaic size is 8x8 pixels.
STUNT_STUNT_SCROLL_LEFT    , // continuously move left
STUNT_STUNT_SCROLL_UP      , // continuously move up
STUNT_STUNT_SCROLL_RIGHT   , // continuously move right
STUNT_STUNT_SCROLL_DOWN    , // continuously move down
ENUM_TEXIAO_NUMBER,
};
#endif

#define FONT_WEIGHT          12
#define WINDOW_FRAME         1
#define WINDOW_FRAME_COLOR   (RGB(0, 0, 255))
#define WINDOW_BACK_COLOR    (RGB(0, 0, 0))
#define FONT_COLOR            (RGB(255, 255, 255))
```