ZH-SDK

The author	Create/Modify Date	Version	Note
Jianbin Du	2018-06-02	V1.0	Basic Link Port

Contents

I. Note

The current SDK only supports the secondary development of full color card software V5.

1). The SDK development platform is Windows7, the development tool is VS2013, the development language is C/C++, the Demo example program development platform is Windows 7, the development tool is VS2013/Qt5.5, and the development language is C/C++;

2). All exported functions are c functions, the calling method is __declspec, Demo support c/c++, Qt5.5.

3). The IP address to be passed is the IP automatically allocated after the asynchronous card is connected to the Network.

II. Basic Function

1. Function: Get SDK version information.

Function Prototype: ZHSDK_API int ZH_SDKVer(void).

Overview: Returns SDK version information.

2. Function: Get the error code of the previous SDK function calling.

Function Prototype: int ZHSDK_API ZH_GetSDKLastError ().

Overview: Returns the error code of the last SDK function call.

3. Function: System Initialization.

Function Prototype:

int ZHSDK_API ZH_Initialize(void *pExParamsBuf, int nBufSize);

Overview:

pExParametersBuf: UI control panel pointer.

nBufSize**:** Reserved parameters, can not be passed in. Return to 1 if successful, return to 0 if failed.

4. Function: Create a screen.

Function Prototype:

int ZHSDK_API ZH_CreateScreen(const std::wstring& name,int width, int height, int backcolor, const std::wstring& imagePath, int nsize);

Overview:

Name: Screen name,

Width: Screen width (0 -1024),

Height: Screen height (0-512),

Backcolor: Screen background color (0x000000-0xffffff),

ImagePath: Screen background image file path,

Nsize: The buffer length of extra parameter ;

Return to the created screen ID (1-n) if successful, return to -1 if failed.

5. Function: Add a program to the screen.

Function Prototype:

int ZHSDK_API ZH_AddProgram(int nScreenID, bool bGlobal, const std::wstring& name, int playmode, const std::string& endTime, int count, int duration, int backcolor, const std::wstring& imagePath, int nsize);

Overview:

NScreenID: Designated screen ID (1-n),

BGlobal: The sign for whether it is an overall program or not (0 or 1),

Name: Screen name,

PlayMode: Play mode (0-4 is the default in order. <When this mode is specified, parameters 5, 6, and 7 are invalid>, and <When this mode is specified, the program will keep playing during playback. Parameters 5, 6, and 7 are invalid>. < When the mode is at the end of the specified time, the parameter 5-endTime is valid, the count and duration of parameter 6-7 are invalid>, and the specified number of times <parameter 6--count is valid, parameter 5, 7 is invalid when the mode is specified number of times> , Specify playback duration <parameter 7-- duration is valid, parameter 5, 6 is invalid when the mode is specified time, PlayDefault

<parameters 5, 6, and 7 are invalid when this mode is specified>),

EndTime: End time(Format: "2018-6-20 20:25:18"),

Count: The number of play times (1-n),

Duration: Duration of the play (1-n, units: second),

BackColor: Program background color (0x000000-0xffffff),

ImagePath: The program background play files,

Nsize: The buffer length of extra parameter ;

Return to program ID (1-n) if successful, return to -1 if failed.

6. Function: Add one partition to the specified program.

Function Prototype:

int ZHSDK_API ZH_AddArea(int nProgramID, const std::wstring& name,int x, int y, int width, int height, bool benable, int leftBorder, int topBorder, int speed, int style, int bordercolor, const std::wstring& imagePath, int iTransparency, int backcolor, std::vector<std::wstring> backgroundlist, int nsize);

Overview:

NProgram ID: The specified program ID (1-n),

Name: The name of the partition.

X: The position coordinate of partition, X (0-n, the width of screen is smaller than the actually created),

Y: The position coordinate of partition, Y (0-n, the height of screen is smaller than actually created),

Width: The width of the partition (0-n, the width of screen is smaller than actually created),

Height: The height of the partition (0-n, the height of screen is smaller than actually created),

Benable: Whether to enable the border (the value of bool),

Left Border: The width of left and right borders (1-10),

Top Border: The width of the upper and lower borders (1-10),

Speed: The speed of Running side (0-10),

Style: Style of border (0-3: static, counterclockwise, clockwise, flashing),

Border Color: Color of border (0x000000-0xffffff),

Image Path: Loading file of border,

ITransparency: The transparency of partition background (0~255),

Back Color: The background color value of partition (0x000000-0xffffff),

Background List: Loading file list of partition background,

Nsize: The buffer length of extra parameter;

Return to the partition ID (1-n) if successful, return to -1 if failed.

7. Function: Add one single-line text material to the specified partition**.**

Function Prototype:

int ZHSDK_API ZH_AddSimpleText(int nAreaID, const std::wstring& name, int times, const std::wstring& content, const std::wstring& Font, bool beBold, bool beItalic, bool beUnderline, int ftSize , int ftWidth, int ftColor, int ftSpace, int moveEffect, int moveSpeed, int duration, int colorMode, std::vector<int> fontColorList, int textureSpeed, int txtStyle, int txtColorTemplate, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int nsize);

Overview:

nAreaID: Specified partition ID (1-n),

Name: The name of single-line text element,

Times: The number of play times (1-100),

Content: Text content,

Font: Font name,

beBol: Whether bold (value of the bool),

beItalic: Whether italic (value of the bool),

beUnderline: whether underline (value of the bool),

ftSize: Font size (8-200, Note: the font size must not exceed the actual partition size)，

ftWidth: Font width (not need temporarily),

ftColor: Font color (0x000000-0xffffff),

ftSpace: Font space(0-99),

moveEffect: Moving mode (402-406: move left, static, move right, move up, move down),

moveSpeed: Moving speed (1-10),

Duration: Duration time (not used),

colorMode: Color mode (reference to Appendix 1),

fontColorList: Color list (used for colorful effects. Note: each element of the list is an RGB value),

textureSpeed: Texture speed (not used),

txtStyle: Text effects (0-3, default words, hollow words, shadow words, three-dimensional words),

txtColorTemplate: Colorful word template (reference to Appendix 2),

Bloop: Whether permanent loop (value of the bool),

enterAnimationEffect: Animation enter effects (not used for single-line text),

enterAnimationDuration: Animation enter time (not used for single-line text),

midAnimationEffect: Animation duration effects (not used for single-line text),

midAnimationDuration: Animation duration time (not used),

leaveAnimationEffect: Animation leave effects (not used for single-line text),

leaveAnimationDuration: Animation leave time (not used for single-line text),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

8. Function: Add one picture material to the specified partition.

Function Prototype: int ZHSDK_API ZH_AddImage (int nAreaID, const std::wstring& name, const std::wstring& path, int fitMode, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int nsize);

Overview:

nAreaID: The specified partition ID (1-n),

Name: The name of picture material,

Path: The path of the image file,

fitMode: The fill mode of image showing in the specified partition (0-4, stretch, original size, width-fit, height-fit, auto-scale),

Bloop: Whether permanent loop (value of the bool),

enterAnimationEffect: Animation Enter effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade in, coverage effect-move down, horizontal shutter effect, vertical shutter effect),

enterAnimationDuration: Animation entry time (1000000-0),

midAnimationEffect: Animation duration effects (0-4, stretch, original size, width-fit, height-fit, auto-scale),

midAnimationDuration: Animation duration time (1000000000-0),

leaveAnimationEffect: Animation leave effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade out, coverage effect-move down, horizontal shutter effect, vertical shutter effect),

leaveAnimationDuration: Animation leave time (1000000-0),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

9. Function: Add one clock material to the specified partition.(Note: This version of the SDK to add clock, the program running on the computer must support OPENGL.)

Function Prototype: int ZHSDK_API ZH_AddClock (int nAreaID, const std::wstring& name, bool bNumberStyle, int shape, bool bSingleLine, POINT position, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int Nsize);

Overview:

nAreaID: Specified partition ID (1-n),

Name: The name of the clock material,

bNumberStyle: Whether to display in digital format (value of the bool),

Shape: Clock shape (0-3, circles, ellipses, squares, rectangles),

bSingleLine: Whether to display in single-line (value of the bool),

Position: The position of the clock center point,

Bloop: Whether permanent loop (value of the bool),

enterAnimationEffect: Animation enter effects (not used),

enterAnimationDuration: Animation entry time (not used),

midAnimationEffect: Animation duration effects (do not use),

midAnimationDuration: Animation duration time (not used),

leaveAnimationEffect: Animation leave effects (not used),

leaveAnimationDuration: Animation leave time (not used),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

4

10. Function: Add one sensor material to the specified partition.

Function Prototype:

int ZHSDK_API ZH_AddSensor (int nAreaID, const std::wstring& name, int types, bool bSingleLine, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int nsize);

Overview:

nAreaID: Specifies partition ID (1-n),

Name: Sensor material name,

Types: Sensor type (0x0000：All types, 0x0001: temperature, 0x0002: humidity, 0x0004: noise, 0x0008: PM2.5, 0x0010: PM10, 0x0020: co2, 0x0040: formaldehyde, 0x0080: wind direction, 0x0100: wind speed, 0x0200: wind power),

bSingleLine: whether to display in single-line (value of the bool),

Bloop: Whether permanent loop (value of the bool),

enterAnimationEffect: Animation enter effects (not used),

enterAnimationDuration: Animation entry time (not used),

midAnimationEffect: Animation stay effects (not used),

midAnimationDuration: Animation duration time (not used),

leaveAnimationEffect: Animation leave effects (not used),

leaveAnimationDuration: Animation leave time (not used),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

6

11. Function: Add one multi-line text material to the specified partition.

Function Prototype:

int ZHSDK_API ZH_AddRichText (int nAreaID, const std::wstring& name, const std::wstring& content, int bgColor, const std::wstring& Font, bool beBold, bool beItalic, bool beUnderline, int ftsize, int ftwidth, int ftcolor, int Ftspace, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int nsize);

Overview:

nAreaID: Specifies partition ID (1-n),

Name: The multi-line text material name,

Content: The content of multiple line text,

bgColor: Background color value of multiple line text (0x000000-0xffffff),

Font: Font name,

beBold: Whether bold (value of the bool),

beItalic: Whether italic (value of the bool),

beUnderline: whether underline (value of the bool),

Ftsize: Font size (8-200),

Ftwidth: Word width (not needed temporarily),

Ftcolor: Font color (0x000000-0xffffff),

Ftspace: Word space (0-99),

Bloop: Whether permanently loop (value of the bool),

enterAnimationEffect: Animation enter effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade out, override effect-move down, horizontal shutters effect, vertical shutters effect),

enterAnimationDuration: Animation enter time (1000000-0), midAnimationEffect: Animation stay effects (not used),

midAnimationDuration: Animation duration time (1000000000-0),

leaveAnimationEffect: Animation leave effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade out, override effect-move down, horizontal shutters effect, vertical shutters effect),

leaveAnimationDuration: Animation leave time (1000000-0),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

12. Function: Add one video material to the specified partition.

Function Prototype:

int ZHSDK_API ZH_AddVideo (int nAreaID, const std::wstring& name, const std::wstring& path, int volume, int bloop, int enterAnimationEffect, int enterAnimationDuration, int midAnimationEffect, int midAnimationDuration, int leaveAnimationEffect, int leaveAnimationDuration, int nsize);

Overview:

nAreaID: Specifies partition ID (0-n),

Name: The video material name,

Path: Path of the video file,

Volume: Video play volume (0-100),

Bloop: Whether permanent loop (value of the bool),

enterAnimationEffect: Animation enter effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade out, override effect-move down, horizontal shutters effect, vertical shutters effect) ,

midAnimationDuration: Animation enter time (1000000-0),

midAnimationEffect: Animation stay effects (not used),

midAnimationDuration: Animation duration time (not used),

leaveAnimationEffect: Animation leave effects (20, 21, 22, 23, 40, 81, 140, 141, move up, move down, move left, move right, fade in and fade out, override effect-move down, horizontal shutters effect, vertical shutters effect),

leaveAnimationDuration: Animation leave time (1000000-0),

Nsize: The buffer length of extra parameter;

Return to self-ID (1-n) if successful, return to -1 if failed.

13. Function: Play the specified project.

Function Prototype:

int ZHSDK_API ZH_PlayProject(void *pDirPath);

Overview:

pDirPath: project path;

Return to 0 if successful, return to -1 if failed.

14. Function: Play the specified program.

Function Prototype:

int ZHSDK_API ZH_PlayAct(int nProgramID);

Overview:

nProgramI: Program ID (1-n);

Return to 0if successful, return to -1 if failed.

15. Function: Play the specified partition.

Function Prototype: int ZHSDK_API ZH_PlayActContent (int nAreaID);

Overview:

nAreaID: Partition ID (1-n); return to 0 if successful, return to -1 if failed.

16. Function Prototype: Play the specified material.

Function Prototype: int ZHSDK_API ZH_PlayMaterial (int nMaterialID);

Overview:

nMaterialID: Material ID (1-n); return to 0 if successful, return to -1 if failed.

17. Function: Get the specified screen position and size.

Function Prototype: Void ZHSDK_API ZH_LedScreenRect (int nScreenID, RECT* rct);

Overview:

nScreenID: Specified screen ID (1-n), rct: RECT pointer;

18. Function: Set position and size of the specified screen.

Function Prototype: void ZHSDK_API ZH_SetLedScreenRect (int nScreenID, const RECT& rct);

Overview:

nScreenID: Specified screen ID (1-n), rct: data of RCT position and size;

19. Function: Delete the specified virtual screen.

Function Prototype: int ZHSDK_API ZH_RemoveScreen (int nScreenID);

Overview: Input the specified screen ID; return to 0 if successful, return to -1 if failed.

20. Function: Delete the specified program.

Function Prototype: int ZHSDK_API ZH_RemoveProgram(int nProgramID);

Overview:

nScreenID: The specified program ID (1-n); return to 0 if successful, return to -1 if failed.

21. Function: Delete the specified partition.

Function Prototype: int ZHSDK_API ZH_RemoveArea(int nAreaID);

Overview:

nAreaID: The specified partition ID (1-n); return to 0 if successful, return to -1 if failed.

22. Function: Delete the specified material.

Function Prototype: int ZHSDK_API ZH_RemoveMaterial(int nMaterialID);

Overview:

nMaterialID: The specifies material ID (1-n); return to 0 if successful, return to -1 if failed.

23. Function: Save the project to the specified contents.

Function prototype: int ZHSDK_API ZH_SaveProject(const std::wstring& pDirPath);

Overview:

pDirPath: The path to save the project. Return to-1 if failed.

24.Function: Send to the specified IP playbox

Function Prototype: int ZHSDK_API ZH_SendToDevice(std::vector<ZH_IP_INFO>* idList);

Overview:

IP list: Playbox basic information ZH_IP_INFO list (ZH_IP_INFO structure contains: Playbox IP address(std::string), user password(std::string), and LED screen ID(int) which controlled by player box); If success then return 0, otherwise return -1.

25. Function: Clear asynchronous program

Function prototype: int ZHSDK_API ZH_ClearProgram(std::vector<ZH_IP_INFO>* idList);

Overview:

IP list: Playbox basic information ZH_IP_INFO list (ZH_IP_INFO structure contains: Playbox IP address(std::string), user password(std::string), and LED screen ID(int) which controlled by player box); If success then return 0, otherwise return -1.

26. Function: Get the qty of LED asynchronous card

Function prototype: int ZHSDK_API ZH_DeviceCount();

Overview: If success, then return the total qty of asynchronous playbox.

27. Function: Set LED asynchronous card power switch

Function prototype: int ZHSDK_API ZH_LedScreenPower(bool powerOn, std::vector<ZH_IP_INFO>* idList);

Overview:

Power on: Whether to turn on the power (bool value: true means on; false means off)

IP list: Playbox basic information ZH_IP_INFO list (ZH_IP_INFO structure contains: Playbox IP address(std::string), user password(std::string), and LED screen ID(int) which controlled by player box. Note: if it’s empty, then turn on/off all asynchronous playbox according to parameter 1); If success then return 0, otherwise return -1.

28. Function: Set LED screen brightness

Function prototype: int ZHSDK_API ZH_SetBrightness(int brightNessValue, std::vector<ZH_IP_INFO>* idList);

Overview:

Brightness value: 0-255

IP list: Playbox basic information ZH_IP_INFO list (ZH_IP_INFO structure contains: Playbox IP address(std::string), user password(std::string), and LED screen ID(int) which controlled by player box. Note: If it’s empty, set the brightness values of all asynchronous playbox to the value passed by parameter 1;

If success then return 0, otherwise return -1.

29. Function: Create an asynchronous card group

Function prototype: int ZHSDK_API ZH_CreateGroup(const std::wstring& name);

Overview:

name: group name; If success then return to the current group ID, otherwise return -1

30. Function: Add an asynchronous card to the group.

Function prototype: int ZHSDK_API ZH_AddOneDeviceToGroup(int groupId,const std::string& ip);

Overview:

groupID: the ID value is the return value for creating the asynchronous card group); IP: asynchronous playbox IP address;

If success then return 0, otherwise return -1.

31. Function: time calibration, contains a single asynchronous card or multiple asynchronous lists or groups.

Function prototype: int ZHSDK_API ZH_CalibrationTime(const std::string& timeArea, const std::string& currentTime,bool sumTm,bool autoSyn,std::vector<ZH_IP_INFO>* idList);

Overview:

Time area: Time zone, incoming value, see Appendix 3 (if the parameter 4 is true, then this parameter is invalid)

Current time: the default is the current system time, if the parameter 4 is true, then this parameter is invalid) (Format："2018-6-20 9:25:18")

sumTm: it's daylight saving time or not (bool value, if the parameter 4 is true, then this parameter is invalid)

autoSyn: whether to automatically synchronize (bool value),

idList: Playbox basic information ZH_IP_INFO list (ZH_IP_INFO structure contains: Playbox IP address(std::string), user password(std::string), and LED screen ID(int) which controlled by player box. Note: If the list is passed in blank, then it represents the time zone of all asynchronous playboxes are set as parameter 1, and the current time is set as parameter 2);

If success then return 0, otherwise return -1.

32. Function: Get device IP list.

Function prototype: int ZHSDK_API ZH_GetDeviceIdList(std::*vector*<ZH_IP_INFO>* idList);

Overview:

idList: Playbox IP and other basic information ZH_IP_INFO list pointer (ZH_IP_INFO structure contains: player IP address, user password, and player box control LED screen ID <Note: This function stores the value of the device is online after this function call, 0 is offline and 1 is online>.

If success then return 0, otherwise return -1.

33. Function: Empty local show.

Function prototype: int ZHSDK_API ZH_ClearLocalActs();

Overview:

After this function is called, all local programs/areas/elements will be cleared.

If success then return 0, otherwise return -1.

34. Function: Cloud server login.

Function prototype: int ZHSDK_API ZH_CloudLogin(const std::*string*& userName, const std::*string*& password);

Overview:

userName： username，

password: password

If success then return 0, otherwise return -1.

35. Function: Cloud Server Logout.

Function prototype: int ZHSDK_API ZH_CloudLogout();

Overview:

Logout cloud server.

If success then return 0, otherwise return -1.

36. Function: The cloud sends programs to asynchronous cards that contain a single asynchronous card or multiple asynchronous lists, or packets.

Function prototype: int ZHSDK_API ZH_CloudSend(const std::wstring& proPath,std::*vector*<std::*string*>* idList);

Overview:

proPath: Local project path,

idList: Cloud device list.

If success then return 0, otherwise return -1.

37. Function: Get a list of server devices and get a list by group.

Function prototype: int ZHSDK_API ZH_GetCloudDeviceIdList(std::*vector*<std::*string*>* idList);

Overview:

idList: Return to cloud device list.

If success then return 0, otherwise return -1.

38. Function: Set the brightness value of the cloud device control screen.

Function prototype: int ZHSDK_API ZH_SetCloudDeviceBrightness(int brightNessValue, std::*vector*<std::*string*>* idList);

Overview:

brightNessValue：Brightness value,

idList：Cloud device list.

If success then return 0, otherwise return -1.

39. Function: Cloud device timing.

Function prototype: int ZHSDK_API ZH_CloudDeviceCalibrationTime(const std::*string*& timeArea, const std::*string*& currentTime, bool sumTm, bool autoSyn, std::*vector*<std::*string*>* idList);

Overview:

Time area: Time zone, incoming value, see Appendix 3 (if the parameter 4 is true, then this parameter is invalid)

Current time: the default is the current system time, if the parameter 4 is true, then this parameter is invalid) (Format："2018-6-20 9:25:18")

sumTm: it’s daylight saving time or not (bool value, if the parameter 4 is true, then this parameter is invalid)

autoSyn: whether to automatically synchronize (bool value),

idList: Cloud device ID list. Note: If the list pointer is empty, the time zone and current time of all cloud devices under the user name are set as parameter 1, parameter 2.

If success then return 0, otherwise return -1.

40. Function: Cloud device switch screen.

Function prototype: int ZHSDK_API ZH_CloudDeviceLedScreenPower(bool powerOn, std::*vector*<std::*string*>* idList);

Overview:

Power on: Whether to turn on the power (bool value: true means on; false means off)

IP list: Cloud device ID list. Note: If the list pointer is passed in as null, the remote screen controlled by all cloud devices under the current user name will be switched according to parameter 1.

If success then return 0, otherwise return -1.

41. Function: Clear cloud device program.

Function prototype: int ZHSDK_API ZH_CloudDeviceClearPrograms(std::*vector*<std::*string*>* idList);

Overview:

idList: Cloud device ID list.

If success then return 0, otherwise return -1.

III. Appendix

1. Color Mode：

0: Default

1: Horizontal gradient 2 colors;

2: Horizontal gradient 3 colors;

3: Horizontal gradient 4 colors;

4: Horizontal gradient 5 colors;

5: Horizontal gradient 6 colors;

6: Vertical gradient 2 colors;

7: Vertical gradient 3 colors;

8: Vertical gradient 4 colors;

9: Vertical gradient 5 colors;

10: Vertical gradient 6 colors;

11: Diagonal gradient 2 colors;

12: Diagonal gradient 3 colors;

13: Diagonal gradient 4 colors;

14: Diagonal gradient 5 colors;

15: Diagonal gradient 6 colors;

16: Anti-diagonal gradient 2 colors;

17: Anti-diagonal gradient 3 colors;

18: Anti-diagonal gradient 4 colors;

19: Anti-diagonal gradient 5 colors;

20: Anti-diagonal gradient 6 colors;

21: Animation gradient 2 colors;

22: Animation gradient 3 colors;

23: Animation gradient 4 colors;

24: Animation gradient 5 colors;

25: Animation gradient 6 colors;

26: Flash 2 colors;

27: Flash 3 colors;

28: Flash 4 colors;

29: Flash 5 colors;

30: Flash 6 colors;

31: Rainbow streamer word (horizontal);

32: Rainbow streamer word (vertical);

33: Rainbow streamer word (diagonal);

34: Rainbow streamer word (anti-diagonal);

35: One word one color, 2 colors;

36: One word one color, 3 colors;

37: One word one color, 4 colors;

38: One word one color, 5 colors;

39: One word one color, 6 colors;

40: 2-color transition;

41: Fade in and fade out;

2. Text Color Template: Colorful Words Template

0: None;

1: Static;

2: 3D rotation word moves up;

3: 3D rotation word moves left;

4: 3D rotation word moves right;

5: Fly in;

6: Explosion 1;

7: Explosion 2(track);

8: Fireworks 1;

9: Fireworks 2(track);

10: Split;

11: Dissolution;

12: Spot;

13: Scan;

14: Monochrome stroke;

15: Light;

16: Swell;

17: Flag fly;

18: Falling;

19: Rise;

20: Approaching;

21: Left roll;

22: Right roll;

23: Wave;

24: Undulation;

25: Push;

26: Clockwise circulation;

27: Anticlockwise circulation;

28: Left move the ring;

29: Right move the ring;

30: Searchlight;

31: Magnifier;

32: Flash;

33: Typing;

34: Ripple;

35: Ambiguous;

36: Mirroring;

37: Rotate X in place;

38: Rotate Y in place;

39: Rotate Z in place;

Note: Asynchronous software doesn’t support colorful words temporarily.

3. Allowed time zone settings:

```
<timezones>
    <timezone id="Pacific/Majuro"> Majuro </timezone>
    <timezone id="Pacific/Midway"> Midway </timezone>
    <timezone id="Pacific/Honolulu"> Honolulu </timezone>
    <timezone id="America/Anchorage"> Anchorage </timezone>
    <timezone id="America/Los_Angeles"> US Pacific Time (Los
Angeles)</timezone>
    <timezone id="America/Tijuana"> Pacific Time (Tijuana)</timezone>
    <timezone id="America/Phoenix"> American Mountain Time (Phoenix)</timezone>
    <timezone id="America/Chihuahua"> Chihuahua </timezone>
    <timezone id="America/Denver"> American Mountain Time (Denver)</timezone>
    <timezone id="America/Costa_Rica"> Central American Time (Costa Rica)</timezone>
    <timezone id="America/Chicago"> Central American Time (Chicago)</timezone>
    <timezone id="America/Mexico_City"> Central American Time (Mexico City)</timezone>
    <timezone id="America/Regina"> Central American Time (Regina)</timezone>
    <timezone id="America/Bogota"> Colombia Time (Bogotá)</timezone>
    <timezone id="America/New_York"> Eastern Time (New York)</timezone>
    <timezone id="America/Caracas"> Venezuela time (Caracas)</timezone>
```

<timezone id="America/Barbados"> Atlantic Time (Barbados)</timezone>

<timezone id="America/Manaus"> Amazon Standard Time (Manaus)</timezone>

<timezone id="America/Santiago"> San Diego </timezone>

<timezone id="America/St_Johns"> Newfoundland Time (St. John's)</timezone>

<timezone id="America/Sao_Paulo"> Sao Paulo </timezone>

<timezone id="America/Argentina/Buenos_Aires"> Buenos Aires </timezone>

<timezone id="America/Godthab"> Gotthorb </timezone>

<timezone id="America/Montevideo"> Uruguay time (Montevideo)</timezone>

<timezone id="Atlantic/South_Georgia"> South Georgia </timezone>

<timezone id="Atlantic/Azores"> Azores </timezone>

<timezone id="Atlantic/Cape_Verde"> Cape Verde </timezone>

<timezone id="Africa/Casablanca"> Casablanca </timezone>

<timezone id="Europe/London"> Greenwich Mean Time (London)</timezone>

<timezone id="Europe/Amsterdam"> Central European Standard Time (Amsterdam)</timezone>

<timezone id="Europe/Belgrade"> Central European Standard Time (Belgrade)</timezone>

<timezone id="Europe/Brussels"> Central European Standard Time (Brussels)</timezone>

<timezone id="Europe/Sarajevo"> Central European Standard Time (Sarajevo)</timezone>

<timezone id="Africa/Windhoek"> Windhoek </timezone>

<timezone id="Africa/Brazzaville"> West African Standard Time (Brazzaville)</timezone>

<timezone id="Asia/Amman"> Eastern European Standard Time (Amman)</timezone>

<timezone id="Europe/Athens"> Eastern European Standard Time (Athens)</timezone>

<timezone id="Asia/Beirut"> Eastern European Standard Time (Beirut)</timezone>

<timezone id="Africa/Cairo"> Eastern European Standard Time (Cairo)</timezone>

<timezone id="Europe/Helsinki"> Eastern European Standard Time (Helsinki)</timezone>

<timezone id="Asia/Jerusalem"> Israeli Time (Jerusalem)</timezone>

<timezone id="Europe/Minsk"> Minsk </timezone>

```
<timezone id="Africa/Harare"> Central African Standard Time (Harare)</timezone>
<timezone id="Asia/Baghdad"> Baghdad </timezone>
<timezone id="Europe/Moscow"> Moscow </timezone>
<timezone id="Asia/Kuwait"> Kuwait </timezone>
<timezone id="Africa/Nairobi"> Eastern African Standard Time (Nairobi)</timezone>
<timezone id="Asia/Tehran"> Iranian Standard Time (Tehran)</timezone>
<timezone id="Asia/Baku"> Baku </timezone>
<timezone id="Asia/Tbilisi"> Tbilisi </timezone>
<timezone id="Asia/Yerevan"> Yerevan </timezone>
<timezone id="Asia/Dubai"> Dubai </timezone>
<timezone id="Asia/Kabul"> Afghanistan time (Kabul)</timezone>
<timezone id="Asia/Karachi"> Karachi </timezone>
<timezone id="Asia/Oral"> Ural </timezone>
<timezone id="Asia/Yekaterinburg"> Yekaterinburg </timezone>
<timezone id="Asia/Calcutta"> Calcutta </timezone>
<timezone id="Asia/Colombo"> Colombo </timezone>
<timezone id="Asia/Katmandu"> Nepal time (Kathmandu)</timezone>
<timezone id="Asia/Almaty"> Almaty </timezone>
```

```
<timezone id="Asia/Rangoon"> Myanmar time (Yangon)</timezone>
<timezone id="Asia/Krasnoyarsk"> Krasnoyarsk </timezone>
<timezone id="Asia/Bangkok"> Bangkok </timezone>
<timezone id="Asia/Shanghai"> China Standard Time (Beijing)</timezone>
<timezone id="Asia/Hong_Kong"> Hong Kong Time (Hong Kong)</timezone>
<timezone id="Asia/Irkutsk"> Irkutsk time (Irkutsk)</timezone>
<timezone id="Asia/Kuala_Lumpur"> Kuala Lumpur </timezone>
<timezone id="Australia/Perth"> Perth </timezone>
<timezone id="Asia/Taipei"> Taipei time (Taipei)</timezone>
<timezone id="Asia/Seoul"> Seoul </timezone>
<timezone id="Asia/Tokyo"> Japan time (Tokyo)</timezone>
<timezone id="Asia/Yakutsk"> Yakutsk time (Yakutsik)</timezone>
<timezone id="Australia/Adelaide"> Adelaide </timezone>
<timezone id="Australia/Darwin"> Darwin </timezone>
<timezone id="Australia/Brisbane"> Brisbane </timezone>
<timezone id="Australia/Hobart"> Hobart </timezone>
<timezone id="Australia/Sydney"> Sydney </timezone>
<timezone id="Asia/Vladivostok"> Vladivostok time
```

```
(Vladivostok)</timezone>
        <timezone id="Pacific/Guam"> Guam </timezone>
        <timezone id="Asia/Magadan"> Magadan Time (Magadan)</timezone>
        <timezone id="Pacific/Auckland"> Auckland </timezone>
        <timezone id="Pacific/Fiji"> Fiji </timezone>
        <timezone id="Pacific/Tongatapu"> East Ghatb </timezone>
</timezones>
```

IV. Demonstration

The demonstration project can be compiled and run with VS2013 demo_c_test.sln in the attached directory:LEDPlayerV5_SDK\demo. The demo program is developed using Qt5.5. There may be a Qt5.5 installation path for PC during development (eg: D:/Qt/Qt5.5.0/5.5/msvc2013/lib). This may be related to debugging. The environment does not match. Please download Qt5.5 installation and modify the project Qt configuration path before use.

0