Logitech Gaming LED SDK

Overview and Reference

Contents

Overview

The Logitech Gaming LED SDK enables applications such as games to control the backlight LEDs on supported Logitech gaming mice and keyboards.
The user has the option to block games from changing the lighting via a setting in the Logitech Gaming Software (version 8.35 and newer). This option is located under the General settings tab of Logitech Gaming Software.
The SDK is a Windows based API for C/C++ programmers. Games based on the Microsoft Win32 API do not access hardware directly. Instead, the Logitech Gaming LED SDK interacts with supported Logitech devices on behalf of the games.
Logitech Gaming Software 8.55+ is required to enable this SDK's features.

SDK Package

The following files are included:

- LogitechLEDLib.h: C/C++ header file containing function prototypes
- LogitechLEDLib.lib: companion lib file to access DLL exported functions (32 and 64 bit)

Requirements

The Logitech Gaming LED SDK can be used on the following platforms:

- Windows XP SP2 (32-bit and 64-bit)
- Windows Vista (32-bit and 64-bit)
- Windows 7 (32-bit and 64-bit)
- Windows 8 (32-bit and 64-bit)

The Logitech Gaming LED SDK is a C based interface and is designed for use by C/C++ programmers. Familiarity with Windows programming is required.

Interfacing with the SDK

Using LogitechLed.h and LogitechLed.lib to access LogitechLed.dll

The application can include LogitechLEDLib.h and link to LogitechLEDLib.lib (see "Sample usage of the SDK" further below or sample program in Samples folder). The lib file loads the dll LogitechLed.dll that ships with Logitech Gaming Software 8.55+, therefore if Logitech Gaming Software is not installed in the host machine, the SDK won't work.

Available colors

Different devices have different capabilities. They range from full single-key RGB support to single color only.
Details for supported devices are found further below in "Features of lighting-capable Logitech Gaming mice and keyboards".
The SDK has a single function to set the backlighting color and takes values for R(ed), G(reen), B(lue). The way it deals with single color devices is to take whichever of the R, G, and B values is the highest and apply it. This is important to remember, because if for example rotating through colors, the game should make sure to alternate the maximum numbers as it rotates so that the effect on a single color device would be noticeable too.

Multiple clients using the SDK at the same time

The SDK allows only one client to control backlighting at any given time. In case two applications try to initialize the SDK, the latest one will take over control.

Features of lighting-capable Logitech Gaming mice and keyboards

G910 Orion Spark

Colors

Single key RGB support. This keyboard supports all the functions available in the SDK, both per-key lighting and full keyboard lighting.

G810 Orion Spectrum

Colors

Single key RGB support. This keyboard supports all the functions available in the SDK, both per-key lighting and full keyboard lighting.

G610 Orion Brown

Colors

Single key Monochrome support. This device accepts all the functions for devices of type LOGI_DEVICETYPE_PERKEY_RGB. It will only display the highest value for R,G,B on each key.

G710+

Colors

Single color only. Full resolution. Highest value for R, G or B defines brightness.

G633 & G933

Colors
Supports full RGB.

G600

Colors
Supports full RGB, will work with the SDK only if set to Host mode through Logitech Gaming Software.

G510/ G510s

Colors
Supports full RGB.

G110

Colors
Supports full R(ed) and B(lue), but not G(reen). When calling the SDK's LogiLedSetLighting function, values for green will be ignored.

G19 / G19s

Colors

Supports full RGB.

G105

Colors

Single color only. Full resolution. Highest value for R, G or B defines brightness.

G105 Call Of Duty

Colors

Single color only. Full resolution. Highest value for R, G or B defines brightness.

G300

Colors

Supports red on/off, green on/off, blue on/off, or a combination of the three. When calling the SDK's LogiLedSetLighting function, if the percentage given is below 50, the color will be off, and when above 50, the color will be on.

G900 Chaos Spectrum

Colors
Supports Full RGB.

G303 Daedalus Apex

Colors
Supports Full RGB.

G11

Colors

Single color only, 3 levels of brightness. When calling the SDK's LogiLedSetLighting function, if the highest RGB percentage given is below 33, the color will be off, if between 33 and 66, the brightness will be low, and when above 66, the brightness will be high.

G13

The SDK treats this device as a keyboard.

Colors

Supports full RGB.

G15 v1

Colors

Single color only, 3 levels of brightness. When calling the SDK's LogiLedSetLighting function, if the highest RGB percentage given is below 33, the color will be off, if between 33 and 66, the brightness will be low, and when above 66, the brightness will be high.

.

G15 v2

Colors

Single color only, 3 levels of brightness. When calling the SDK's LogiLedSetLighting function, if the highest RGB percentage given is below 33, the color will be off, if between 33 and 66, the brightness will be low, and when above 66, the brightness will be high.

Do's and Don'ts

These are a few guidelines that may help you implement 'better' support in your game:

- If you don't use the LogiLedSetTargetDevice function, remember that some devices have only a single color. They will work fine if flashing a red warning light for example (their color will flash), but if rotating lighting try to make sure that the max value of the three colors goes up and down so that single color devices will have their brightness go up and down.
- Whenever doing a temporary lighting effect, do not forget to save the current lighting (using LogiLedSaveCurrentLighting function) just before starting the effect, and then restoring the lighting (via SDK's LogiLedRestoreLighting function) right after the effect is finished. This only applies to user defined effects, the saving-restore lighting is already included in the preset effects (LogiLedFlashLighting and LogiLedPulseLighting).
- When calling LogiLedSetLighting, Logitech Gaming Software will make sure to not override current brightness for devices that only support single color. Therefore, setting the lighting to 100% red, on a G710+ it will result in a max brightness according to the user hardware settings.

Sample usage of the SDK

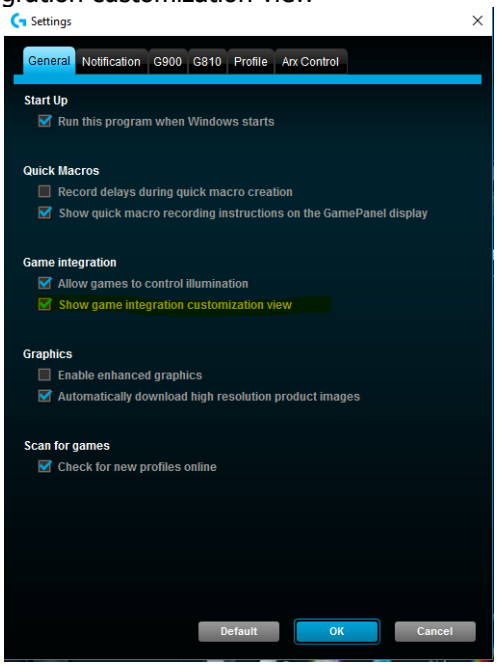
```
#include " LogitechLEDLib.h"

…

LogiLedInit();
// Be sure to do other things to give some time before calling LogiLedSetLighting()

…

// Save current lighting before starting some temporary effect
LogiLedSaveCurrentLighting();

…

int red = …;
int green = …;
int blue = …;

LogiLedSetLighting(red, green, blue);


…
// Call per-key lighting effects
LogiLedSetLightingForKeyWithKeyName(keyboardNames::ARROW_DOWN, red, green, blue);

…
// Possibly call effect functions
LogiLedFlashLighting(red, green, blue, duration, interval);
…

LogiLedPulseLighting(red, green, blue, duration, interval);
```

```
// Restore previously saved lighting when effect is finished
LogiLedRestoreLighting();

…

LogiLedShutdown();
```

Reference

ConfigOption Functions

The **LogiLedGetConfigOption** function set, allows the developer to query for an option set by the user and use that value to customize the interaction with the SDK. A call to any of these functions will create an entry in the Logitech Gaming Software – Applet Manager View. This view is disabled by default, since it's something targeting only "Advanced users", to enable it click on the Settings Icon in LGS and then check the box "Show Game integration customization view"




```
bool LogiLedGetConfigOptionNumber(wchar_t *configPath, double *defaultValue);

bool LogiLedGetConfigOptionBool(wchar_t *configPath, bool *defaultValue);

bool LogiLedGetConfigOptionColor(wchar_t *configPath, int *defaultRed, int
*defaultGreen, int *defaultBlue);

bool LogiLedSetConfigOptionLabel(wchar_t *configPath, wchar_t *label);
```

Parameters

- **configPath**: This identifies the option uniquely. This can be just a string (E.G. “Terrorist”) or it can be a two level tree (“Colors/Terrorist”). If the two level tree is specified, the option will be displayed in Logitech Gaming Software as an entry (“Terrorist”) inside a group (“Colors”).
- **defaultValue**: This parameter, depending on the specific function takes the default value for the relative option. If the option has been modified through LGS by the user, it will be filled in with the modified value, otherwise the default value will be saved (to be shown to the user) and it won’t be modified.

Return value
The function always returns true, unless some bad parameter has been specified.

Usage Example

```
double healthFlashingThreshold = 0.15;
LogiLedGetConfigOptionNumber(L"player/flashing_edge", &healthFlashingThreshold);
//This healthFlashingThreshold value will now contain the option as set by the user,
or the default value if it has never been set.

//This function is just to display a prettier name in the LGS customization interface.
LogiLedSetConfigOptionLabel(L"player/flashing_edge", L"Flash Health Percentage");

if(player.health() < healthFlashingThreshold)
{
	LogiLedFlashLighting(100, 0, 0, 0, 100);
}
```

LogiLedInit
The **LogiLedInit**() function makes sure there isn't already another instance running and then makes necessary initializations. It saves the current lighting for all connected and supported devices.
This function will also stop any effect currently going on the connected devices.

```
bool LogiLedInit();
```

Return value
If the function succeeds, it returns true. Otherwise false.
If it returns false, means that the connection with Logitech Gaming Software is broken, make sure that it is running.

LogiLedGetSdkVersion
The **LogiLedGetSdkVersion** () function retrieves the version of the SDK version installed on the user's system.

```
bool LogiLedGetSdkVersion(int *majorNum, int *minorNum, int *buildNum);
```

Parameters
- majorNum: [in] the function will fill this parameter with the major build number of the sdk installed in the system
- minorNum: [in] the function will fill this parameter with the minor build number of the sdk installed in the system
- buildNum: [in] the function will fill this parameter with the build number of the sdk installed in the system

Return value
If the function succeeds, it returns true. Otherwise false.

If it returns false, means that there is no SDK installed on the user system, or the sdk version could not be retrieved.

LogiLedSetTargetDevice

The **LogiLedSetTargetDevice** () function sets the target device type for future calls. The default target device is LOGI_DEVICETYPE_ALL, therefore, if no call is made to LogiLedSetTargetDevice the SDK will apply any function to all the connected devices.

```
bool LogiLedSetTargetDevice(int targetDevice);
```

Parameters

- targetDevice: one or a combination of the following values:

```
LOGI_DEVICETYPE_MONOCHROME
LOGI_DEVICETYPE_RGB
LOGI_DEVICETYPE_PERKEY_RGB
LOGI_DEVICETYPE_ALL
```

Return value

If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called, the parameter is wrong, or if the connection with Logitech Gaming Software was lost.

Example

```
LogiLedInit();
LogiLedSetTargetDevice(LOGI_DEVICETYPE_RGB | LOGI_DEVICETYPE_MONOCHROME);
//From now on the calls to LED SDK will only affect RGB and MONOCHROME devices, PER_KEY
devices such as G910 will ignore this calls
LogiLedSetLighting(100,0,0);
…
LogiLedSetTargetDevice(LOGI_DEVICETYPE_PERKEY_RGB);
//Future calls will only affect per-key rgb devices such as G910.
LogiLedSetLightingForKeyWithKeyName(keyboardNames::ARROW_DOWN, 100, 0, 0);
LogiLedFlashLighting(50, 50, 50, 0, 300);
…
LogiLedSetTargetDevice(LOGI_DEVICETYPE_ALL);
//From now on we'll affect all the connected devices
LogiLedSetLighting(50, 0, 0);
…
LogiLedShutDown();
```

LogiLedSaveCurrentLighting

The **LogiLedSaveCurrentLighting**() function saves the current lighting so that it can be restored after a temporary effect is finished. For example if flashing a red warning sign for a few seconds, you would call the **LogiLedSaveCurrentLighting**() function just before starting the warning effect.
On per-key backlighting supporting devices, this function will save the current state for each key.

```
bool LogiLedSaveCurrentLighting();
```

Return value

If the function succeeds, it returns true. Otherwise false.

The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSetLighting

The **LogiLedSetLighting**() function sets the lighting on connected and supported devices.

```
bool LogiLedSetLighting(int redPercentage, int greenPercentage, int bluePercentage);
```

Parameters

- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.
- bluePercentage: amount of blue. Range is 0 to 100.

Return value

If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

Remarks

Do not call this function immediately after LogiLedInit(). Instead leave a little bit of time after LogiLedInit().
For devices that only support a single color, the highest percentage value given of the three colors will define the intensity. For monochrome backlighting device, Logitech Gaming Software will reduce proportionally the value of the highest color, according to the user hardware brightness setting.

LogiLedRestoreLighting

The **LogiLedRestoreLighting**() function restores the last saved lighting. It should be called after a temporary effect is finished. For example if flashing a red warning sign for a few seconds, you would call this function right after the warning effect is finished.
On per-key backlighting supporting devices, this function will restore the saved state for each key.

```
bool LogiLedRestoreLighting();
```

Return value

If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedFlashLighting

The **LogiLedFlashLighting** () function saves the current lighting, plays the flashing effect on the targeted devices and, finally, restores the saved lighting.

```
bool LogiLedFlashLighting (int redPercentage, int greenPercentage, int bluePercentage,
int milliSecondsDuration, int milliSecondsInterval);
```

Parameters

- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.

- bluePercentage: amount of blue. Range is 0 to 100.
- milliSecondsDuration : duration of the effect in milliseconds, this parameter can be set to `LOGI_LED_DURATION_INFINITE` to make the effect run until stopped through **LogiLedStopEffects()**
- milliSecondsInterval : duration of the flashing interval in milliseconds

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called, if the connection with Logitech Gaming Software was lost or if another effect is currently running.

LogiLedPulseLighting
The **LogiLedPulseLighting** () function saves the current lighting, plays the pulsing effect on the targeted devices and, finally, restores the saved lighting.

```
bool LogiLedPulseLighting(int redPercentage, int greenPercentage, int bluePercentage, int
milliSecondsDuration, int milliSecondsInterval);
```

Parameters
- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.
- bluePercentage: amount of blue. Range is 0 to 100.
- milliSecondsDuration : duration of the effect in milliseconds, this parameter can be set to `LOGI_LED_DURATION_INFINITE` to make the effect run until stopped through **LogiLedStopEffects()**
- milliSecondsInterval : duration of the flashing interval in milliseconds

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called, if the connection with Logitech Gaming Software was lost or if another effect is currently running.

LogiLedStopEffects
The **LogiLedStopEffects** () function stops any of the presets effects (started from LogiLedFlashLighting or LogiLedPulseLighting).

```
bool LogiLedStopEffects();
```

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSetLightingFromBitmap
The **LogiLedSetLightingFromBitmap** () function, sets the array of bytes passed as parameter as colors to per-key backlighting featured connected devices.

```
bool LogiLedSetLightingFromBitmap(unsigned char bitmap[]);
```

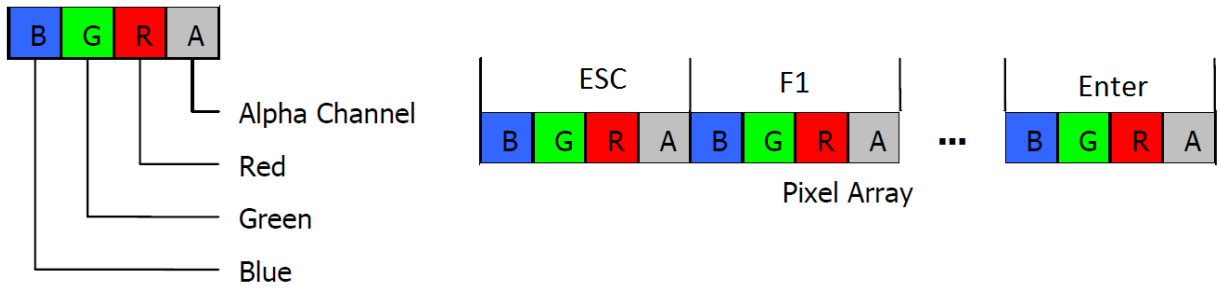
Parameters

- bitmap: a unsigned char array containing the colors to assign to each key on the per-lighting device connected. The size required for this bitmap is defined by `LOGI_LED_BITMAP_SIZE`

The array of pixels is organized as a rectangular area, 21x6, representing the keys on the device. Each color is represented by four consecutive bytes (RGBA).
Here is a graphical representation of the bitmap array:

byte 0-3 ESC	byte 4-7 F1	byte 8-11 F2	…	byte 72-75 NULL	byte 76-79 NULL	byte 80-83 NULL
byte 84-87 `	byte 88-91 1	byte 92-95 2	…	byte 156-159 /	byte 160-163 *	byte 164-167 -
…	…	…	…	…	…	…
byte 420-423 CTRL	byte 424-427 WIN	byte 428-431 ALT	…	byte 495-498 NUM0	byte 499-502 ./DEL	byte 500-503 NULL

32 bit values are stored in 4 consecutive bytes that represent the RGB color values for that pixel. These values use the same top left to bottom right raster style transform to the flat character array with the exception that each pixel value is specified using 4 consecutive bytes. The illustration below shows the data arrangement for these RGB quads.



Each of the bytes in the RGB quad specify the intensity of the given color. The value ranges from 0 (the darkest color value) to 255 (brightest color value).

Return value

If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

Remarks

The array passed in has to be allocated by the caller of the size `LOGI_LED_BITMAP_SIZE`. If the array is smaller, the function will apply the effect to a smaller portion of the keyboard and set everything else to black. If the array is bigger, the remaining part will be ignored. To create partial bitmaps and update only

parts of the keyboard, set the alpha channel for the keys to ignore to 0. This will allow to update just portion of the keyboard, without overriding the other keys.

LogiLedExcludeKeysFromBitmap

The **LogiLedExcludeKeysFromBitmap** () function sets a list of keys, defined by keynames to be ignored when calling the function LogiLedSetLightingFromBitmap. This is useful when creating effects on the bitmap during gameplay loop, but still wanting to set some keys on top of that using the LogiLedSetLightingFromKeyName.

```
bool LogiLedExcludeKeysFromBitmap (LogiLed::KeyName *keyList, int listCount);
```

Parameters

- keyList: A preallocated array of LogiLed::KeyName(s) to be excluded when calling LogiLedSetLightingFromBitmap
- listCount: the number of items in the list KeyList

LogiLedSetLightingForKeyWithScanCode

The **LogiLedSetLightingForKeyWithScanCode** () function sets the key identified by the scancode passed as parameter to the desired color. This function only affects per-key backlighting featured connected devices.

```
bool LogiLedSetLightingForKeyWithScanCode (int keyCode, int redPercentage, int
greenPercentage, int bluePercentage);
```

Parameters

- keyCode: the scan-code of the key to set
- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.
- bluePercentage: amount of blue. Range is 0 to 100.

Return value

If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSetLightingForKeyWithHidCode

The **LogiLedSetLightingForKeyWithHidCode** () function sets the key identified by the hid code passed as parameter to the desired color. This function only affects per-key backlighting featured connected devices.

```
bool LogiLedSetLightingForKeyWithHidCode (int keyCode, int redPercentage, int
greenPercentage, int bluePercentage);
```

Parameters

- keyCode: the hid-code of the key to set
- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.

- bluePercentage: amount of blue. Range is 0 to 100.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSetLightingForKeyWithQuartzCode
The **LogiLedSetLightingForKeyWithQuartzCode** () function sets the key identified by the quartz code passed as parameter to the desired color. This function only affects per-key backlighting featured connected devices.

```
bool LogiLedSetLightingForKeyWithQuartzCode (int keyCode, int redPercentage, int
greenPercentage, int bluePercentage);
```

Parameters
- keyCode: the quartz-code of the key to set
- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.
- bluePercentage: amount of blue. Range is 0 to 100.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSetLightingForKeyWithKeyName
The **LogiLedSetLightingForKeyWithKeyName** () function sets the key identified by the code passed as parameter to the desired color. This function only affects per-key backlighting featured connected devices.

```
bool LogiLedSetLightingForKeyWithHidCode (LogiLed::KeyName keyCode, int redPercentage,
int greenPercentage, int bluePercentage);
```

Parameters
- keyCode: one of the key codes from the enum `KeyName`:
  - `ESC = 0x01,`
  - `F1 = 0x3b,`
  - `F2 = 0x3c,`
  - `F3 = 0x3d,`
  - `F4 = 0x3e,`
  - `F5 = 0x3f,`
  - `F6 = 0x40,`
  - `F7 = 0x41,`
  - `F8 = 0x42,`
  - `F9 = 0x43,`
  - `F10 = 0x44,`
  - `F11 = 0x57,`

- F12 = 0x58,
- PRINT_SCREEN = 0x137,
- SCROLL_LOCK = 0x46,
- PAUSE_BREAK = 0x45,
- TILDE = 0x29,
- ONE = 0x02,
- TWO = 0x03,
- THREE = 0x04,
- FOUR = 0x05,
- FIVE = 0x06,
- SIX = 0x07,
- SEVEN = 0x08,
- EIGHT = 0x09,
- NINE = 0x0A,
- ZERO = 0x0B,
- MINUS = 0x0C,
- EQUALS = 0x0D,
- BACKSPACE = 0x0E,
- INSERT = 0x152,
- HOME = 0x147,
- PAGE_UP = 0x149,
- NUM_LOCK = 0x145,
- NUM_SLASH = 0x135,
- NUM_ASTERISK = 0x37,
- NUM_MINUS = 0x4A,
- TAB = 0x0F,
- Q = 0x10,
- W = 0x11,
- E = 0x12,
- R = 0x13,
- T = 0x14,
- Y = 0x15,
- U = 0x16,
- I = 0x17,
- O = 0x18,
- P = 0x19,
- OPEN_BRACKET = 0x1A,
- CLOSE_BRACKET = 0x1B,
- BACKSLASH = 0x2B,
- KEYBOARD_DELETE = 0x153,
- END = 0x14F,
- PAGE_DOWN = 0x151,
- NUM_SEVEN = 0x47,
- NUM_EIGHT = 0x48,
- NUM_NINE = 0x49,
- NUM_PLUS = 0x4E,
- CAPS_LOCK = 0x3A,

- A = 0x1E,
- S = 0x1F,
- D = 0x20,
- F = 0x21,
- G = 0x22,
- H = 0x23,
- J = 0x24,
- K = 0x25,
- L = 0x26,
- SEMICOLON = 0x27,
- APOSTROPHE = 0x28,
- ENTER = 0x1C,
- NUM_FOUR = 0x4B,
- NUM_FIVE = 0x4C,
- NUM_SIX = 0x4D,
- LEFT_SHIFT = 0x2A,
- Z = 0x2C,
- X = 0x2D,
- C = 0x2E,
- V = 0x2F,
- B = 0x30,
- N = 0x31,
- M = 0x32,
- COMMA = 0x33,
- PERIOD = 0x34,
- FORWARD_SLASH = 0x35,
- RIGHT_SHIFT = 0x36,
- ARROW_UP = 0x148,
- NUM_ONE = 0x4F,
- NUM_TWO = 0x50,
- NUM_THREE = 0x51,
- NUM_ENTER = 0x11C,
- LEFT_CONTROL = 0x1D,
- LEFT_WINDOWS = 0x15B,
- LEFT_ALT = 0x38,
- SPACE = 0x39,
- RIGHT_ALT = 0x138,
- RIGHT_WINDOWS = 0x15C,
- APPLICATION_SELECT = 0x15D,
- RIGHT_CONTROL = 0x11D,
- ARROW_LEFT = 0x14B,
- ARROW_DOWN = 0x150,
- ARROW_RIGHT = 0x14D,
- NUM_ZERO = 0x52,
- NUM_PERIOD = 0x53,
- G_1 = 0xFFF1,
- G_2 = 0xFFF2,

- G_3 = 0xFFF3,
- G_4 = 0xFFF4,
- G_5 = 0xFFF5,
- G_6 = 0xFFF6,
- G_7 = 0xFFF7,
- G_8 = 0xFFF8,
- G_9 = 0xFFF9,
- G_LOGO = 0xFFFF1,
- G_BADGE = 0xFFFF2,

- redPercentage: amount of red. Range is 0 to 100.
- greenPercentage: amount of green. Range is 0 to 100.
- bluePercentage: amount of blue. Range is 0 to 100.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedSaveLightingForKey
The **LogiLedSaveLightingForKey** () function saves the current color on the keycode passed as argument. Use this function with the LogiLedRestoreLightingForKey to preserve the state of a key before applying any effect.
This function only applies to device of the family LOGI_DEVICETYPE_PERKEY_RGB.

```
bool LogiLedSaveLightingForKey(LogiLed::KeyName keyName)
```

Parameters
- keyName: The key to save the color for. A value from the LogiLed::KeyName enum.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedRestoreLightingForKey
The **LogiLedRestoreLightingForKey** () function restores the saved color on the keycode passed as argument. Use this function with the LogiLedSaveLightingForKey to preserve the state of a key before applying any effect.
This function only applies to device of the family LOGI_DEVICETYPE_PERKEY_RGB.

```
bool LogiLedRestoreLightingForKey(LogiLed::KeyName keyName)
```

Parameters
- keyName: The key to restore the color on. A value from the LogiLed::KeyName enum.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedFlashSingleKey
The **LogiLedFlashSingleKey** () function starts a flashing effect on the key passed as parameter. The key will be flashing with an interval as defined by msInterval for msDuration milliseconds, alternating the color passed in as parameter and black. This function only applies to device of the family LOGI_DEVICETYPE_PERKEY_RGB.

```
bool LogiLedFlashSingleKey(LogiLed::KeyName keyName, int redPercentage, int
greenPercentage, int bluePercentage, int msDuration, int msInterval)
```

Parameters
- keyName: The key to restore the color on. A value from the LogiLed::KeyName enum.
- redPercentage : amount of red in the active color of the flash effect. Range is 0 to 100.
- greenPercentage : amount of green in the active color of the flash effect. Range is 0 to 100.
- bluePercentage : amount of blue in the active color of the flash effect. Range is 0 to 100.
- msDuration : duration in milliseconds of the effect on the single key. This parameter can be set to `LOGI_LED_DURATION_INFINITE` to make the effect run until stopped through **LogiLedStopEffects()** or **LogiLedStopEffectsOnKey()**

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedPulseSingleKey
The **LogiLedPulseSingleKey** () function starts a pulsing effect on the key passed as parameter. The key will be pulsing with from start color to finish color for msDuration milliseconds. This function only applies to device of the family LOGI_DEVICETYPE_PERKEY_RGB.

```
bool LogiLedPulseSingleKey(LogiLed::KeyName keyName, int startRedPercentage, int
startGreenPercentage, int startBluePercentage, int finishRedPercentage, int
finishGreenPercentage, int finishBluePercentage, int msDuration, bool isInfinite);
```

Parameters
- keyName: The key to restore the color on. A value from the LogiLed::KeyName enum.
- startRedPercentage: amount of red in the start color of the pulse effect. Range is 0 to 100.
- startGreenPercentage: amount of green in the start color of the pulse effect. Range is 0 to 100.
- startBluePercentage: amount of blue in the start color of the pulse effect. Range is 0 to 100.
- finishRedPercentage amount of red in the finish color of the pulse effect. Range is 0 to 100.
- finishGreenPercentage: amount of green in the finish color of the pulse effect. Range is 0 to 100.
- finishBluePercentage: amount of blue in the finish color of the pulse effect. Range is 0 to 100.
- msDuration: duration in milliseconds of the effect on the single key.

- isInfinite : if this is set to true the effect will loop infinitely until stopped with a called to **LogiLedStopEffects()** or **LogiLedStopEffectsOnKey()**

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedStopEffectsOnKey
The **LogiLedStopEffectsOnKey** () function stops any ongoing effect on the key passed in as parameter. This function only applies to device of the family LOGI_DEVICETYPE_PERKEY_RGB.

```
bool LogiLedStopEffectsOnKey(LogiLed::KeyName keyName);
```

Parameters
- keyName: The key to stop the effects on. A value from the LogiLed::KeyName enum.

Return value
If the function succeeds, it returns true. Otherwise false.
The function will return false if **LogiLedInit**() hasn't been called or if the connection with Logitech Gaming Software was lost.

LogiLedShutdown
The **LogiLedShutdown** () function restores the last saved lighting and frees memory used by the SDK.

```
void LogiLedShutdown();
```

End-User License Agreement for Logitech Gaming LED SDK

This End-User License Agreement for Logitech Gaming LED SDK ("Agreement") is a legal agreement between you, either an individual or legal entity ("You" or "you") and Logitech Inc. ("Logitech") for use of the Logitech Gaming LED software development kit, which includes computer software and related media and documentation (hereinafter "Logitech Gaming LED SDK"). By using this Logitech Gaming LED SDK, you are agreeing to be bound by the terms and conditions of this Agreement. If you do not agree to the terms and conditions of this Agreement, promptly return the Logitech Gaming LED SDK and other items that are part of this product in their original package, or if you have downloaded this software from a Logitech or a Distributor web site, then you must stop using the software and destroy any copies of the software in your possession or control.

1 **Grant of License and Restrictions.** This Agreement grants You the following rights provided that You comply with all terms and conditions of this Agreement.

(a) Logitech grants You a limited, non-exclusive, nontransferable license to install and use an unlimited number of copies of the Logitech Gaming LED SDK on computers. All other rights are reserved to Logitech.

(b) You shall not reverse engineer, decompile or disassemble any portion of the Logitech Gaming LED SDK, except and only to the extent that this limitation is expressly prohibited by applicable law.

(c) At your option, you may provide reasonable feedback to Logitech, including but not limited to usability, bug reports and test results, with respect to the Logitech Gaming LED SDK. All bug reports, test results and other feedback provided to Logitech by You shall be the property of Logitech and may be used by Logitech for any purpose.

(d) In the event Logitech, in its sole discretion, elects to provide copies of the Logitech Gaming LED SDK to more than one individual employed by You (if You are not a single individual), each such individual shall be entitled to exercise the rights granted in this Agreement and shall be bound by the terms and conditions herein.

2 **Updates.** Logitech is not obligated to provide technical support or updates to You for the Logitech Gaming LED SDK provided to You pursuant to this Agreement. However, Logitech may, in its sole discretion, provide further pre-release versions, technical support, updates and/or supplements ("Updates") to You, in which case such Updates shall be deemed to be included in the "Logitech Gaming LED SDK" and shall be governed by this Agreement, unless other terms of use are provided in writing by Logitech with such Updates.

3 **Intellectual Property Rights.** The Logitech Gaming LED SDK is licensed, not sold, to You for use only under the terms and conditions of this Agreement. Logitech and its suppliers retain title to the Logitech Gaming LED SDK and all intellectual property rights therein. The Logitech Gaming LED SDK is protected by intellectual property laws and international treaties, including U.S. copyright law and international copyright treaties. All rights not expressly granted by Logitech are reserved.

4 **Disclaimer of Warranty.** TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, LOGITECH, ITS SUPPLIERS AND DISTRIBUTORS PROVIDE THE LOGITECH GAMING LED SDK AND OTHER LOGITECH PRODUCTS AND SERVICES (IF ANY) AS IS AND WITHOUT WARRANTY OF ANY KIND. LOGITECH AND ITS SUPPLIERS AND DISTRIBUTORS EXPRESSLY DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS WITH RESPECT TO THE LOGITECH GAMING LED SDK AND ANY WARRANTIES OF NON-INTERFERENCE OR ACCURACY OF INFORMATIONAL CONTENT. NO LOGITECH DISTRIBUTOR, AGENT, OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATION, EXTENSION, OR ADDITION TO THIS WARRANTY. Some jurisdictions do not allow limitations on how long an implied warranty lasts, so the above limitation may not apply to you.

5 **Limitation of Liability.** IN NO EVENT WILL LOGITECH, ITS SUPPLIERS, OR DISTRIBUTORS BE LIABLE FOR ANY COSTS OF PROCUREMENT OF SUBSTITUTE PRODUCTS OR SERVICES, LOST PROFITS, LOSS OF INFORMATION OR DATA, OR ANY OTHER SPECIAL, INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING IN ANY WAY OUT OF THE SALE OF, USE OF, OR INABILITY TO USE THE LOGITECH GAMING LED SDK OR ANY LOGITECH PRODUCT OR SERVICE, EVEN IF LOGITECH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO CASE SHALL LOGITECH'S, ITS SUPPLIERS' AND DISTRIBUTORS' TOTAL LIABILITY EXCEED THE ACTUAL MONEY PAID FOR THE LOGITECH PRODUCT OR SERVICE GIVING RISE TO THE LIABILITY. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you. The above limitations will not apply in case of personal injury where and to the extent that applicable law requires such liability.

6 **U.S. Government Rights.** Use, duplication, or disclosure of the software contained in the Logitech Gaming LED SDK by the U.S. Government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988) FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable. Logitech Inc. 7600 Gateway Blvd, Newark, CA 94560.

7 **Export Law Assurances.** You agree and certify that neither the Logitech Gaming LED SDK nor any other technical data received from Logitech will be exported outside the United States except as authorized and as permitted by the laws and regulations of the United States. If you have rightfully obtained the Logitech Gaming LED SDK outside of the United States, you agree that you will not re-export the Logitech Gaming LED SDK nor any other technical data received from Logitech, except as permitted by the laws and regulations of the United States and the laws and regulations of the jurisdiction in which you obtained the Logitech Gaming LED SDK.

8 **Termination:** This Agreement is effective until terminated. Upon any violation of any of the provisions of this Agreement, or any provisions of any agreement between you and a Distributor, rights to use the Logitech Gaming LED SDK shall automatically terminate and the Logitech Gaming LED SDK must be returned to Logitech or all copies of the Logitech Gaming LED SDK destroyed. You may also terminate this Agreement at any time by destroying all copies of the Logitech Gaming LED SDK in your possession or control. If Logitech makes a request via public announcement or press release to stop using the copies of the Logitech Gaming LED SDK, you will comply immediately with this request. The provisions of paragraphs 3, 7, 8 and 12 will survive any termination of this Agreement.

9 **General Terms and Conditions.** If You are an individual signing this Agreement on behalf of a company, then You represent that You have authority to execute this Agreement on behalf of such company. This Agreement will be governed by and construed in accordance with the laws of the United States and the State of California, without regard to or application of its choice of law rules or principles. If for any reason a court of competent jurisdiction finds any provision of this Agreement, or portion thereof, to be unenforceable, that provision of the Agreement shall be enforced to the maximum extent permissible so as to affect the intent of the parties, and the remainder of this Agreement shall continue in full force and effect. This Agreement constitutes the entire agreement between You and Logitech respect to the use of the Logitech Gaming LED SDK and supersedes all prior or contemporaneous understandings, communications or agreements, written or oral, regarding such subject matter.