Q Light Controller+ on the Raspberry Pi

User guide

Last updated: November 7th, 2015

1. Introduction

Welcome to the QLC+ on the Raspberry Pi user guide !

A Raspberry Pi runs a Linux operating system, so please don't expect to find all the tools and windows of a desktop system like Windows or OSX.

Also, please keep in mind that a Raspberry Pi cannot replace a PC, especially when it comes to designing a light show.

This document is not meant to teach you how to use Linux. There's plenty of information about that on the internet, as well as specific information on how to setup and tweak a Raspberry Pi Linux system.

So please, before asking non QLC+ related questions, submit them to Google and not in the QLC+ forum.

2. Features

QLC+ on Raspberry Pi is an optimized version that tries to get the best out of a limited device while keeping the whole set of functionalities offered on the desktop version.

Let's go directly into the techy details to let you understand what kind of gear is powering QLC+:

- A Linux system based on Debian Wheezy, with a 4.1.6 kernel and optimized for the ARM (hard float) platform
- Supports Raspberry Pi 1 model B/B+ and Raspberry Pi 2 model B
- Around **360MB of memory available** on RPi1 Model B/B+ when QLC+ is running. **850MB free** on RPi2 Model B.
- Extremely tiny software image that fits into a SD card with a size of just 1GB and still leaves around **360MB of free space**
- Powered by Qt 5.5.1 **directly running on OpenGL** for fast 2D rendering of the User Interface. Xorg is not present in the image.
- All the QLC+ plugins are available in this version, including OLA and a powerful **SPI plugin to natively control RGB LED Panels**
- **Pi1 model B/B+ boots in just 30 seconds, Pi2 model B in 22 seconds.** They start automatically in operate mode and with web access activated. It is possible to define a project to be loaded and started at boot time.

3. Download & install on a SD card

Starting to play with QLC+ on your Raspberry Pi requires a couple of quick and easy steps.

1. Download the software image from the URL provided via email. You need to use a download manager supporting file transfer resuming, like the Chrome's Chrono Download Manager extension, jDownloader or simply the 'wget' command.

2. Once you have downloaded the software, you will notice that it is compressed into a 7z archive, to save you time and space. The 7z format is extremely space-saving but it requires an additional tool to extract it.

   Here's how to obtain it for different platforms:

   - **Windows**: The official tool can be downloaded from here: http://www.7-zip.org/
   - **Mac OSX**: An unofficial tool can be downloaded from here: https://www.macupdate.com/app/mac/19139/ez7z
   - **Linux**: depending on your distribution, you might find a 7z extractor already installed. On Ubuntu you need p7zip (sudo apt-get install p7zip-full). Once installed, it should already be integrated with Ark, Nautilus, Gnome Archive Manager, and so on.

   Just double click the file if unsure. Once extracted, you should obtain a file with a name like this: `qlcplus_raspbian_wheezy_20151021.img`

3. To copy the .img file on your SD card, you obviously need a SD card at least 1GB big. **Please keep in mind that this operation will fully erase your SD card, so backup your data first if there is any !**

   On Windows and OSX you need additional tools to write the software in the SD card. In Linux you should already have everything you need.

   Here's an extensive guide to help you through this operation. Please follow the chapter named "Flashing the SD card using ...” for your operating system.

   http://elinux.org/RPi_Easy_SD_Card_Setup

   Please be very careful in doing this operation as you can loose important data of your computer !

4. Raspberry Pi connections

Depending on your needs, the Raspberry Pi itself might not be enough to control your lights.

- **[USB]** If you're going to use USB devices, such as DMX adapters or MIDI-USB controllers, then you should be aware that the Raspberry Pi 1 USB ports are limited in current to 200mA, so they won't be enough to support some kind of devices. In this case it is suggested to **connect a powered USB HUB** that can be found for a few $/€ on Amazon or eBay.

  You can connect a mouse and a keyboard without a HUB.

  The Raspberry Pi 2 does not have limitations on the USB ports. Just choose wisely the power adapter you're going to use. Depending on the needs, 5V/1,5A or 5V/2A should be OK.

- **[HDMI]** If you're going to use QLC+ like on your PC, then you will need all the above plus a TV set, connected to the HDMI connector (or the analog one)

  Please, don't expect to have the same performances of a PC, as the Raspberry Pi is not a PC. Keep in mind that version 1 has just a single core 700MHz CPU, while version 2 has a quad core 900MHz CPU.

- **[Ethernet]** If you're going to use the ArtNet, E1.31 or OSC plugins, then you just need to plug an ethernet cable to your Raspberry and make sure a DHCP server (usually present in traditional routers) assigns a valid IP address to it.

  You can also use static IP addresses, for example if your hosts are connected just with an ethernet hub or directly with a cross cable.

- **[WiFi]** WiFi connection is possible through USB dongles. Additional tweaks are needed to make them work. In general, here's a list of verified devices that are known to be working with the Raspberry Pi:

  http://elinux.org/RPi_USB_Wi-Fi_Adapters

  http://elinux.org/RPi_VerifiedPeripherals

  The 3 chipsets known to be working OK on the Raspberry Pi are: Broadcom BCM43143, Realtek 8188 and Ralink 3070

- **[SPI]** If you're going to connect some LED strips or a RGB panel on the SPI port, then you need to have a custom cable that you can't find on the market. Usually it's very easy to solder it, but it all depends on how the signals are mapped on your strips. The Raspberry PINs used to control a SPI device are mapped as in **picture A**.

  Basically you need to use PINs 19, 21, 23 and 25. At the moment, the QLC+ SPI plugin doesn't control CE (Chip Enable) signals.

- **[GPIO]** If you're going to use the GPIO plugin, please be aware that not all the RPi IO PINs can be used as GPIO. Refer to **picture A** to understand the correct numbers you should configure in the QLC+ GPIO plugin.

  Note also that since GPIO numbering start from 0 and QLC+ DMX channels start from 1, there is a +1 delta you need to remember. For example GPIO.18 will control DMX channel 19.

It should be possible to have more GPIO available by unloading the SPI and I^{2}C kernel modules but for now this is out of the scope of this guide.

Picture A

26 PIN IO RPi 1 model B Rev 1			
Name	**Pin #**		**Name**
3.3V	1	2	**5V**
I^{2}C SDA.1	3	4	**5V**
I^{2}C SCL.1	5	6	**GND**
1-Wire	7	8	**UART** Tx
GND	9	10	**UART** Rx
GPIO.17	11	12	**GPIO**.18
GPIO.21	13	14	GND
GPIO.22	15	16	**GPIO**.23
3.3V	17	18	**GPIO**.24
SPI MOSI	19	20	GND
SPI MISO	21	22	**GPIO**.25
SPI SCLK	23	24	**SPI** CE0
GND	25	26	**SPI** CE1

26 PIN IO RPi 1 model A/B Rev 2			
Name	**Pin #**		**Name**
3.3V	1	2	**5V**
I^{2}C SDA.1	3	4	**5V**
I^{2}C SCL.1	5	6	**GND**
1-Wire	7	8	**UART** Tx
GND	9	10	**UART** Rx
GPIO.17	11	12	**GPIO**.18
GPIO.27	13	14	GND
GPIO.22	15	16	**GPIO**.23
3.3V	17	18	**GPIO**.24
SPI MOSI	19	20	GND
SPI MISO	21	22	**GPIO**.25
SPI SCLK	23	24	**SPI** CE0
GND	25	26	**SPI** CE1

40 PIN IO RPi 1 model B+ and RPi 2			
Name	**Pin #**		**Name**
3.3V	1	2	**5V**
I^{2}C SDA.1	3	4	**5V**
I^{2}C SCL.1	5	6	**GND**
1-Wire	7	8	**UART** Tx
GND	9	10	**UART** Rx
GPIO.17	11	12	**GPIO**.18
GPIO.27	13	14	GND
GPIO.22	15	16	**GPIO**.23
3.3V	17	18	**GPIO**.24
SPI MOSI	19	20	GND
SPI MISO	21	22	**GPIO**.25
SPI SCLK	23	24	**SPI** CE0
GND	25	26	**SPI** CE1
I^{2}C SDA.0	27	27	**I^{2}C** SCL.0
GPIO.5	29	30	**GND**
GPIO.6	31	32	**GPIO**.12
GPIO.13	33	34	**GND**
GPIO.19	35	36	**GPIO**.16
GPIO.26	37	38	**GPIO**.20
GND	39	40	**GPIO**.21

- **[UART]** If you purchased a BitWizard DMX interface, be aware that by design the hardware is configured for DMX Input. To switch it to output mode you need to add the following lines to the file `/etc/init.d/qlcplus`, right before the line starting with `QLCPLUS_OPTS=”...”`:

```
echo 18 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio18/direction
echo 1 > /sys/class/gpio/gpio18/value
```

Those 3 commands will actually raise GPIO 18 and the BitWizard interface will start emitting DMX signal.

5. Booting and remote access

Before booting your Raspberry Pi, make sure the following requirements are met:

- SD Card with QLC+ software is correctly inserted in the Pi SD card slot
- Micro USB power plug is firmly plugged to the Pi
- If you're using network plugins, make sure the Ethernet cable is correctly plugged in and the Pi can reach the target devices. If you're using a wireless network, make sure your USB WiFi dongle is plugged in an powered with enough current
- If you're using USB adapters, make sure they're correctly plugged to a powered USB HUB, providing enough current to make them work
- if you're using a SPI connection, make sure the cable is plugged to the right Pi PINs as described above

Once all the preliminary checks are done, you are ready to turn your Raspberry Pi on.

After around 30 seconds, you will be able to reach it on the network.

By factory default, the QLC+ Raspberry software is configured with the following IP address: **192.168.0.252**

Hint: In general, to know which IP address your Raspberry is using, just connect a mouse and a TV to it and when QLC+ is started, go to the Input/Output panel and see which output the ArtNet and the E1.31 plugins are displaying.

The first time you need to access your Raspberry Pi, the factory IP address needs to be used. There are three ways to access the device: Web interface, direct access or SSH.

5.1 Web Interface

By default, the QLC+ Raspberry software starts up with a web interface enabled.

It can be accessed from any modern web browser running on any device, such as a computer, a tablet or a smartphone. Your browser need to support the web sockets technology to communicate with QLC+. It is recommended to use Google Chrome.

To access the QLC+ web interface simply connect to this address:

http://<IP address>:9999

Where <IP address> is the Raspberry Pi address. For example the factory web interface will be located at http://192.168.0.252:9999

The web interface allows you to remotely control QLC+ and the Raspberry Pi system without the need to connect a TV set, thus making the system light and portable.

On a Raspberry Pi, the web interface consists in three pages:

- QLC+ Virtual Console
- QLC+ Configuration
- Raspberry Pi System Configuration

5.1.1 QLC+ Virtual Console

This page is displayed by default when accessing the web interface address and it represents the QLC+ Virtual Console.

If a project is loaded, this page will display the widgets previously created with QLC+, otherwise it will be just an empty page.

It is possible to load a project with the “**Load project**” button placed on the top left corner of the page. A window will show up, allowing you to choose a file from the device you're using to control the Raspberry Pi. The project file will be transferred via web and loaded by QLC+ already in operate mode, thus ready to be used immediately.

To access the QLC+ configuration page, just click on the “**Configuration**” button.

5.1.2 QLC+ Simple Desk

Like in the QLC+ desktop version, the web interface exposes a simple desk page to quickly control single channels. This can be particularly useful when checking the DMX mapping of each fixture.

Please note that the web Simple Desk is not as powerful as the desktop version. It is just a simplified tool to help users in some circumstances.

5.1.3 QLC+ Configuration

This page allows to remotely set the QLC+ configuration, divided in three areas:

- **Universes configuration**: Allows to set the inputs, outputs, feedback, profiles and passthrough mode for each QLC+ universe. This is basically the same functionality of the QLC+ input/output panel.

  Since a QLC+ project contains also the I/O information, most likely you won't have to manually configure it on this page, but just check if everything is correct.
- **Audio configuration**: Allows to select the audio devices used for audio playback or audio input.
- **User loaded fixtures**: Allows to remotely load a custom fixture to

QLC+. When clicking on the “**Load fixture**” button, a window will show up, allowing you to choose a file from the device you're using to control the Raspberry Pi. The fixture file will be transferred via web and loaded by QLC+.

When adding new custom fixtures it is recommended to reload a project or either reboot the Raspberry Pi

Once the configuration has been set, it is possible to go back to the web interface main page by clicking on the “**Back**” button, placed at the top left corner of the page.

When clicking on the “**System**” button, instead, it is possible to access the Raspberry Pi system configuration page.

5.1.4 Raspberry Pi System Configuration

This page has been specifically designed for the Raspberry Pi and it's not present on the desktop version of QLC+.

Here it is possible to do the following operations:

- **Network configuration**: A list of the available network interfaces will be displayed here, showing how they are currently configured.

  It is possible to configure wired and wireless interfaces by setting automatic or manual IP address, subnet mask and default gateway.

  In case of wireless networking, it is also possible to set the Access Point name (SSID) and the access password. WPA-PSK is the only supported method.

  When done, click on the “**Apply changes**” button. A popup message will inform you that the changes will take effect at the next boot.
- **Project autostart**: Here it is possible to set the currently loaded project to automatically start up when QLC+ starts. This functionality, in conjunction with the QLC+ “Function autostart” feature, will turn your Raspberry Pi into a completely standalone device. For example if you set a Chaser to play in loop mode, you can have a 24/7 light show very easily.

  When done, click on the “**Apply changes**” button. A popup message will inform you that the changes will take effect at the next boot.

  Once changes are applied, the currently loaded project will be copied into a file in the /root/.qlcplus folder called “autostart.qxw”.

  At the next boot, QLC+ will detect the presence of the file and will load it automatically.
- **Reboot**: this button simply reboots the Raspberry Pi
- **Shutdown**: this button initiate the shutdown sequence

The “**Back**” button places at the top left corner of the page will lead you back to the Virtual Console page.

5.2 Direct Access

If you connect a TV to your Raspberry Pi through an HDMI or analog cable, you will notice that QLC+ is running in fullscreen, with the whole UI as in the desktop version.

If you have a mouse and a keyboard plugged in, you can control QLC+ like the desktop version, but with limited performances due to the reduced CPU power.

You can either purchase some HDMI+Touchscreen solutions available on the market to control the Virtual Console just with your fingers.

It is not suggested to do project editing directly on the Raspberry Pi, but just live control.

To gain control of the Raspberry Pi Linux system while QLC+ is running, just hit the keys CTRL+ALT+Backspace to obtain a shell access. From there login as "**`root`**" with password "**`raspberry`**".

5.3 SSH Access

This type of connection is for expert users. You don't need to worry about this if you don't need to manually make changes to the QLC+ Raspberry software.

The ssh terminal command is normally present in Linux and OSX, while on Windows you will need an extra tool like PuTTY: http://www.putty.org/

The SSH access is possible with the 'root' user like this:

```
ssh root@<IP address>
```

Where <IP address> is the Raspberry Pi address. For example the factory SSH access will be: **`ssh root@192.168.0.252`**

The root user password is: **`raspberry`**

Once the access is granted, you will have a standard Linux prompt. From here you can do everything you want but it is required a bit of experience with the Linux/Debian system.

QLC+ is a standard service following the Debian rules. You can start/stop it by typing

```
service qlcplus stop/start/restart
```

If you need the OLA server to be automatically started at boot just type:

```
dpkg-reconfigure ola
```

6. Date and time

Unlike a desktop computer, the Raspberry Pi doesn't have a backup battery, so when you turn it off, all the system temporary information gets lost.

Most of them are detected during the boot process, but the system date and time is strictly related to the network presence.

If no network is available, the system won't be able to set the correct date and time, so if your QLC+ project uses time schedules, keep in mind that you will need a network access. In this case “network access” means access to a NTP server, which is normally available with an internet connection, so make sure the Raspberry Pi has access to it.

If you're an advanced user and know how to set up a NTP server in your local network, there will be no need for an internet access, but you'll have to manually configure the Raspberry Pi to pick up the date and time from your own server.

7. Locales

In Linux terms, “locales” are international codes (e.g. en_US, de_DE, es_ES, etc) that define the system language.

To change the image system language, just type:

```
dpkg-reconfigure locales
```

And follow the instructions on the screen.

You can also change the layout of your keyboard, if needed, by typing:

```
dpkg-reconfigure keyboard-configuration
```

8. Overscan

Depending on the TV set or touchscreen device connected to the Pi, it might be useful to enable or disable the overscan feature.

When disabled, QLC+ will use every pixel of the display. When enabled, QLC+ will be displayed with a 5% black margin to fit some displays.

By default the overscan feature is enabled at boot.

To disable it, you need to manually modify the QLC+ startup script located at: **`/etc/init.t/qlcplus`**

Look for a line starting with “**`QLCPLUS_OPTS`**” and remove the “**`--overscan`**” option.

9. Displays and touchscreens

Connecting a display with a touchscreen surface is possible in most of the cases, and in the last years a wide variety of products spread out in the market.

However, this topic is not for beginners, so **do not order random displays from random chinese websites unless you know exactly what you're doing and how those devices work !**

Here, some information are provided to help users to work with displays and touchscreens, but it is impossible to cover all the cases due to the wide variety of devices mentioned above.

If the instructions here do not work for you, it's up to your Linux skills to adapt them to your device and you are very welcome to share your findings in the QLC+ Raspberry Pi forum.

If you're not familiar with editing files with Linux, kernel drivers, bash scripts and environment variables, then this chapter is not for you. Sorry.

First of all, you should be aware that there are mainly 3 types of displays that can be connected to the Raspberry Pi:

- HDMI + touchscreen
- VGA + touchscreen
- SPI + touchscreen

They use totally different connections and totally different drivers.

The display output connections and the touchscreen connections are now split into separate paragraphs.

9.1. HDMI displays

With HDMI displays, it's like connecting a TV, so you should see the QLC+ interface out of the box. If you don't, then you most likely need to adjust the RPi HDMI resolution, which doesn't depend on QLC+, but is plenty described here: http://elinux.org/RPiconfig#Video_mode_options

9.2. VGA displays

Since the RPi doesn't have a VGA connector, an adapter is required to make traditional monitors to work with it.

Normally you can find quite cheap VGA-to-HDMI passive adapters on eBay or similar.

Most likely you will have to adjust the screen resolution, with the same instructions for a HDMI display. For example these parameters in the config.txt file might work for you:

```
disable_overscan=1
hdmi_drive=1
hdmi_group=2
hdmi_mode=16
config_hdmi_boost=4
```

9.3. SPI displays

This is the most “embedded” way to connect a display to the Raspberry Pi, and probably the one that needs most skills.

SPI is a serial interface, so display based on this protocol can't be very large. You can find 1.8”, 2.5” and 3.5” sizes (and probably others)

The interesting thing is that they normally provide a single connector covering the whole RPi IO connector and carrying the touchscreen data as well.

Thanks to the awesome FBTFT project (you are invited to read most of the wiki pages) the support for SPI displays is already integrated in the mainline RPi kernel.

There are several drivers, each with several configuration parameters, so here the basic information is provided to give a rough idea of how a SPI display can be used with QLC+.

Edit 'config.txt' file in the vfat partition and add this line at the end of the file:

```
dtoverlay=piscreen,speed=16000000,rotate=90
```

if '`piscreen`' doesn't work for you (e.g. you don't see any image on the TFT, the possible values are:

`hy28a, hy28b, mz61581, piscreen, pitft28-resistive (former pitft), rpi-display, tinylcd35`

Also, you might want to play with the “**`speed`**” and the “**`rotate`**” parameters.

(***optional***) If you want to see the Linux boot messages, edit the 'cmdline.txt' file and add this at the end of the kernel parameters:

```
fbcon=map:10
```

Boot the RPi, login via SSH and test the screen with this command:

```
cp /dev/urandom /dev/fb1
```

If this command shows random pixels on the screen, then it means you are using the right kernel driver and Linux has been able to map the display on the `/dev/fb1` device. Congratulations !

Now, to finally redirect QLC+ to the fb1 device, you need to adjust its command line options.

Edit the file **`/etc/init.d/qlcplus`** and look for a line starting with **`QLCPLUS_OPTS=”...”.`**

Edit the parameters to have something like this:

```
QLCPLUS_OPTS="-platform linuxfb:fb=/dev/fb1 -plugin
evdevtouch:invertx --web -operate"
```

The “**evdevtouch**” plugin has several options you can try:

“invertx” “inverty” “rotate=90”

When you're done, restart the qlcplus service (or reboot the RPi) and you should see QLC+ running on your display with touchscreen support !

9.4. Touchscreens

A touchscreen surface is nothing but a thin film that provides spatial and pressure information. It can be resistive or capacitive and it can have different types of connections.

The easiest way QLC+ can work with a touchscreen, is if Linux is able to map it to a /dev/input/event device, so the preliminary test to do is to check (without any mouse or keyboard plugged) if the following command produces any result:

```
ls -l /dev/input/event*
```

If you don't get any result, then you will need to look up for additional information concerning your specific touchscreen chipset/connection.

If, instead the touchscreen is mapped on some event device, you can test if it's working properly by issuing the following command and then touching the screen:

```
apt-get install evtest
evtest /dev/input/event0
```

Obviously you need to replace “`event0`” with the name of the device actually mapped by Linux and displayed by the `ls` command.

Some touchscreens provide a **USB** cable, and they most likely will work out of the box, as Linux can map a USB input device like if it was a mouse or a keyboard.

If they don't work, then they probably use a non conventional chipset. If the chipset is recognized by Linux, then you could find some information by googling the chipset name, otherwise you might need to give up and consider it “not supported”.

If the touchscreen provides a **RS232** serial connection, then you might encounter hard times to make it work. One possible way could be using the 'tslib', which is supported by the Qt libraries and should support the serial interface.

The parameters you can play with are an 'export' and a Qt specific command line option. For example these commands might work for you:

```
export TSLIB_DEVICE=/dev/ttyS0
export TSLIB_FBDEVICE=/dev/fb0
qlcplus -plugin tslib --web -operate
```

The tslib has also some tools like "ts_calibrate" and "ts_test" to check if the touchscreen is working. They can be installed with

```
apt-get install libts-bin
```

If the previously mentioned "evtest" works for you, then you don't need to use the tslib plugin, so please do not mix the two things.

In case you connected a SPI display, the touchscreen is probably using some **GPIO** signals and the FBTFT kernel driver should make the work also to map a `/dev/input/event` device.

You just need to tell the Qt libraries that you are using a touchscreen and not a mouse with the option:

```
-plugin evdevtouch
```

(see paragraph 7.3 for more details)

Some more Qt-specific information to configure input devices are documented here: http://doc.qt.io/qt-5/embedded-linux.html

However, it has already been discovered that sometimes that documentation is not complete and you will need to dig into the Qt forums to learn more.

10.Frequently Asked Questions

- **Q: What is the .md5 file I've been provided with ?**

  A: MD5 is a widely used technique to provide an integrity checksum of a file, especially if it is big. It's a bit-to-bit hash number calculated on the original file data.

  When the file is downloaded, it is possible to recalculate the MD5 and check if it is equal to the one calculated “offline”. It they don't match, it means the file got corrupted during the download process.

  In Linux and Mac OSX, a command line tool called “md5sum” is available for the purpose. In Windows you need an extra tool like for example **WinMD5**.

- **Q: The user interface is slow on my Raspberry Pi 1**

  A: On a Raspberry Pi 1, the software image is definitely not designed for project editing, so it is suggested to use the software only via web interface or use the RPi connected to a TV only for Virtual Console live operations

- **Q: I moved my SD card on another Raspberry and I have no network interface**

  A: after the first boot, a file is created in the SD card, binding the Linux system to the network interface MAC address. If you move the SD on another RPi, the eth0 interface won't be there. You'll find an inactive eth1. To solve this just reflash the software image or delete the file with the following command and then reboot:

  ```
  rm /etc/udev/rules.d/70-persistent-net.rules
  ```

- **Q: I need more space in my SD card for additional files/packages**

  A: From a SSH console, launch the `raspi-config` tool and select

  ```
  1 Expand Filesystem
  ```

- **Q: When I type `apt-get install …` I always get '`No package found`'**

  A: That's because to save some SD card space, the cache of the apt packages has been removed. To re-create it just type:

  ```
  apt-get update
  ```

- **Q: I use my Raspberry with no ethernet cable (or with a static IP address) and the boot is very slow**

  A: When switching the network configuration from dynamic to static IP, it might happen that the DNS entries don't get updated correctly.

Edit the file

```
/etc/resolv.conf
```

and write a single nameserver entry like this

```
nameserver 127.0.0.1
```

Once done, save and reboot.

- **Q: Every window of the user interface don't have a title bar, so I cannot move them or in some cases close them**

  A: This is a very well known issue and unfortunately a proper solution hasn't been found yet.

  Most likely, a new image will be provided in the future with a light window manager included.

  If you can't live without this feature, you need to download a standard Raspbian image (which includes an X Server and a window manager) and build QLC+ from sources on your own. Please note that the performances in this case can be slightly different from the official image.

- **Q: How do I access the data of my USB pendrive ?**

  A: USB mass storage automount is possible with an extra package that can be installed in this way:

  ```
  apt-get install usbmount
  ```

  Once installed, your pendrive can be accessed from one of the **/media/usbX** folders