

OmplayerGUI Manual

Content

1) Introduction.....	2
2) Installation and Usage.....	3
3) omxplayerGUI front end.....	4
4) omxaudioplayer.....	5
5) omxplayerGUI – the video player window.....	6
a) Understanding the two different modes.....	6
b) Using the interface.....	6
c) Keyboard control	8
d) Windowless Mode.....	9
7) omxplayerGUI settings.....	10
a) Choosing presets.....	10
b) List of all options used by omxplayerGUI.....	10
c) Creating presets.....	14
d) Editing the settings directly.....	14
Appendix A: Keyboard commands.....	15
Special OmxplayerGUI Keyboard Controls.....	15
omxplayer(GUI) Keyboard Controls (Play Mode).....	16
Appendix B: Known problems.....	17
Appendix C: How to Install a Working Version of Youtube-dl.....	18

1) Introduction

omxplayerGUI is a python program to manage playing all kinds of media with omxplayer in lots of different ways. It's one of the helper scripts of Minimal Kiosk Browser 1.6, but it can also be used standalone. omxplayerGUI also supports extracting videos from websites containing HTML5 video or from all video websites supported by youtube-dl. It uses Python's Tkinter GUI elements and so does not require any other packages to be installed.

Version 1.4 of Minimal Kiosk Browser and the accompanying kwebhelper.py script already contained a new GUI to play audio files and whole playlists, called "omxaudioplayer". In version 1.5 of Minimal Kiosk Browser this has been enhanced to a simple, but effective GUI to play also videos inside a window. Of course this is a "fake" to some degree, as omxplayer is still not aware of any windows and simply overlays the video area above the X-Windows-Screen. But new dbus commands have made it possible to resize and move this video area while the video is playing and "omxplayer GUI" makes use of these new options. This is only possible if you use the newest versions of omxplayer from <http://omxplayer.sconde.net/>. Older versions are also supported, but in a more limited way (see below for details).

Omxplayergui.py has been separated from kwebhelper.py and can now also be used as a standalone application from the desktop. If it is started without any arguments, a simple front end will be shown.

Both kwebhelper.py and omxplayergui.py still share the same global settings file kwebhelper_settings.py. But configuration is much easier now from a settings configuration page within Minimal Kiosk Browser.

2) Installation and Usage

omxplayerGUI is installed as part of the Minimal Kiosk Browser (kweb) package and this is the recommended way, even if you are not interested in using the web browser itself, because it's the only way to edit the rather complex settings of omxplayerGUI in a user friendly way.

After installation you will find omxplayerGUI as a new program in your applications menu. From the command line you can run it with:

```
omxplayergui.py [option] [mode] [url] [mimetype]
```

“option” may be one of those:

“--preset=presetname” or “--config=fullpath”,

where “presetname” must be the name of a preset file created with kweb's settings page and “fullpath” must be the full path to a python file (.py), that contains global variables, similar to kwebhelper_settings.py

mode may be either 'av' or 'ytdl' and url is any valid file path or web url. If no mode is given, 'av' will be used. 'ytdl' is the extraction mode for web video and requires a valid URL to a web page containing web video. OmxplayerGUI uses youtube-dl to extract video links from websites like youtube.com and many others. It can also extract HTML5 video links directly.

You can also right click any media (audio, video or playlist) file inside your file manager window to open it with omxplayerGUI.

If no argument is given or called from the application menu, omxplayerGUI will start with a simple front end.

For web video support, you have to install youtube-dl. Currently the version from the Raspbian repository doesn't work correctly any more. Appendix C will show you alternate ways to get a really working version of youtube-dl.

Standalone version

If you want to install omxplayerGUI without Minimal Kiosk Browser, you have to download the source distribution of Minimal Kiosk Browser. It contains a script that only installs omxplayerGUI and the things it needs. Run it with

```
sudo /installomxplayergui
```

To remove it again, run

```
sudo ./removeomxplayergui
```

Configuration of the standalone version of the program is only possible by directly editing the python file:

/usr/local/bin/kwebhelper_settings.py (as root).

3) omxplayerGUI front end

When not used from Minimal Kiosk Browser and started without any arguments, omxplayerGUI will open with a very simple front end.



If you click the “Open” button, a simple file dialogue will appear and you may select any kind of media file to play. Depending on the kind of the file, either omxaudioplayer or omxplayerGUI (the video player) will open and start playing your file(s).

It should be noted, that Python's Tkinter file dialogue is very simple and doesn't look very nice. Minimal Kiosk Browser's “Open” command can be used instead and uses a much more sophisticated GTK+2/3 file selector.

The “Play/Stream” and “Extract” functions require a valid URL or file path, that you have to enter in the text field above. You can use copy & paste for this, by right clicking into the text field, holding the mouse key pressed and selecting “clear & paste” from the small pop-up menu.

If you have a valid (full) file path or a web link URL pointing to a audio/video file or stream on the internet (or a playlist) entered into the text field, you can play it by clicking on Play/Stream. Depending on the content, either the audio or video player interface will be opened.

If you have entered an URL pointing to a web video page (something like “<http://www.youtube.com/watch?v=o774LMtfeJY>”), you can click the “Extract” button. Youtube-dl will be called to extract the video URL from that link and, if successful, omxplayerGUI will open and start playing the video. Youtube-dl is not very fast on the Raspberry Pi and it may take between 8 and 18 seconds, before the video window appears. You can also use links to web pages containing HTML5 video and then omxplayerGUI will extract the video URLs from such pages and play them.

There's also a special function included to search for videos on youtube.com. If you enter something like:

`?raspberrypi`

into the text entry field and click on “Extract”, a search on youtube.com for these key words will be started and after a while (around one minute!), a video window will appear and start playing the first video of the search results. If you stop the video, you will see a playlist with up to 20 search results for “raspberrypi”, all of which can be directly played without any further delay. If the last of the search terms is a number, it will be used to set the page number of the youtube search, e.g.

`?raspberrypi 3`

will result in page3 (videos number 40 - 59) of a youtube search for “raspberrypi.”

The “Edit Settings” button is only available, if you have installed Minimal Kiosk Browser. If you click on it, it will open a browser window as root (password may be required) and show the settings page for kwebhelper and omxplayerGUI (see below for details).

After playing any kind of media and closing the player window, you will always return to the omxplayerGUI front end. Click on “Quit” or the close symbol of the window to close it.

4) omxaudioplayer

If you open an audio file or a playlist (m3u and pls) containing only audio files or streams, omxplayerGUI will be opened in audio mode as “omxaudioplayer”. All files will be shown in a playlist.



Buttons and their keyboard controls:

Play/Pause – Space, p, Return or Keypad Enter: will play the currently selected song, or pause or resume playing, when it is already playing.

Stop – ESC or q: will stop playing a song.

Rewind - ←: Jump backwards by about 10 seconds.

Forward - →: Jump forward by about 10 seconds.

Previous – ↑: If a song is playing and selected, stop it and play the previous song in the list. If a song is playing and you have selected another song, jump to that song and play it. If no song is playing, select the previous song and play it.

Next – ↓: If a song is playing and selected, stop it and play the next song in the list. If a song is playing and you have selected another song, jump to that song and play it. If no song is playing, select the next song and play it.

Volume control: move the slider to change the volume between -60 and +12 db. You can also click into the grey areas besides the slider to change the volume by 3 db, or use the '-' and '+' keys (also on the numeric keypad).

The keyboard commands are very similar to those used by omxplayer, except for the Up and Down arrow keys, which are not used for jumping ahead or backwards by 10 minutes, but for playing the previous or next song.

Functions inside the playlist window:

If you select another song, while a song is playing, it will be played, when the current song is finished or when you click on the Previous or Next button. If you double click another song than the currently selected one, it will be played (and a currently playing song will be stopped). If you double click the currently playing song, it will be stopped.

5) omxplayerGUI – the video player window

a) Understanding the two different modes

Depending on a setting in kwebhelper_settings.py (edited via kweb's settings page),

freeze_window = False

= extended mode, or

freeze_window = True

= simple mode

two different modes can be used.

In simple mode (without using dbus commands) a video window is "frozen" to the screen while a video is playing. The video window cannot be moved, resized or minimized until the video has been stopped. Although this seems to be a kind of restriction, it has also some advantages: it works with older (< 0.3.5) versions of omxplayer and does not use the new dbus commands, which still have some bugs at the moment. The use of simple mode is shown in the title bar of the window with an extension: "(f)" (= frozen).

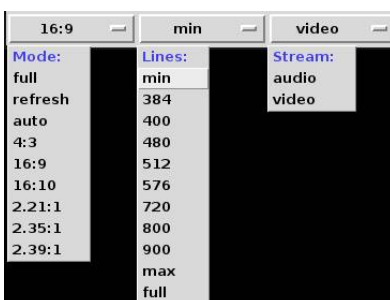
In extended mode the video window can be moved, resized or minimized while a video is playing. The video area will follow within a second, with a slight "hiccup". While the window is minimized, the video will be invisible but play on and you will still hear the sound. You can also seek in the video by moving the slider to a definite position.

Even in extended mode, the video window is "frozen" for a few seconds at the beginning (shown in the title bar) and some controls are disabled, while the program checks, if seeking in the video is possible and if its duration is known. If seeking is not possible, the window remains in the frozen state. If a duration cannot be found, the seek bar remains disabled. It's also possible to toggle between both modes at run time with a keyboard command ALT+u (if video has been stopped).

Some videos (especially rtmp://, rtsp:// and mmsh:// streams) will always be played in frozen or simple mode, if they do not support seeking.

You can play more than one video at the same time. I've managed to play up to 4 video at the same time, 3 SD video and one 720p video TV stream. This shows how powerful the GPU of the RPi is. If you play more than one video and pause one of them, the sound will get lost on all videos. You have to pause and start again each video to get the sound back (this may change with newer versions of omxplayer, see issue <https://github.com/popcornmix/omxplayer/issues/184>). It's also possible to overlay videos using different layers (see keyboard commands below)

b) Using the interface



When running in video mode, the GUI window has two lines of control elements at the bottom. The first line is the same as with omxaudioplayer (see above for details). The second line contains three menu buttons and another slider, which will be explained in detail here. In simple mode, you can only change these settings when no video is running; in extended mode most of these settings are available while a video is being played.

It's also possible, to hide those controls completely (and use only keyboard commands to control the player window). This can be toggled at run time with ALT+h. You can start the player in this mode by setting "hide_controls" to "True" (see below)

The "**Mode:**" menu contains the following choices:

full - play video full screen without any window elements

refresh - play video full screen without any window elements with omxplayer's -r option to change screen resolution and frequency

Both of these options have only an effect, when no video is playing. You have to select full screen mode, before you start a video.

auto - try to get information about the aspect ratio of the video and set the play area accordingly inside the current window. This will take another two or three seconds, before the video starts - on large avi files even more. This works both in simple and extended mode.

If the video has been started in one of the other numbered modes, in extended mode auto detection may run in the background and will be available after a short while, so you can switch to auto mode when the selected mode doesn't fit. This behaviour has to be enabled in kwebhelper_settings (it takes time and processing power and sometimes does not even work). Only available in extended mode.

If you are using a recent version of omxplayer it will support the "aspect" dbus command. This is now also supported by omxplayerGUI (in extended mode)). In this case you can switch to auto mode a few seconds after the video has been started and the video will be displayed with the correct aspect ratio.

4:3, 16:9, 16:10, 2.21:1, 2.35:1 , 2.39:1

All these options define the ratio of width to height of a video manually. In extended mode, you can change these settings, while a video is running, in simple mode only when a video has been stopped or not started yet. If none of these ratios seems to be right, choose auto mode (in extended mode also at run time, if enabled).

The "**Lines:**" menu contains a number of settings that define the height of the video area, ranging from

"**min**" (minimal height, see below)

then some numbers like that:

320,384,400,480,512,576,720,800,900

to

max (maximize window)

and

full (full screen without window title bar, but controls at the bottom)

The "min" size depends on the font height used for the interface, set in kwebhelper_settings.py. The smallest height is 288 lines, when the font height is set to the minimal value of 10 (points). In all other cases it's font height*28. The numbers following "min" depend on the min value and on the size of your screen. In extended mode you can change the height of the video at run time and the

video area will be resized immediately. It's also possible to resize the video window manually by dragging the corner of the window.

The "Stream:" menu contains only two options, "video" and "audio". It can only be changed, if the player has been stopped. In some cases omxplayerGUI cannot decide, if a stream link provides audio (e. g. web radio) or video. You can select if the player should handle such streams as audio or video streams. It's not really important in most cases, as omxplayer will handle it by itself, but you can avoid getting a black screen (when full or refresh mode has been selected) when you just want to listen to a web radio stream.

If the player window is large or near the bottom of the screen, the menu list will pop up upwards and may be hidden behind the video area (if a video is running); there's no way to avoid this because of the way omxplayer works. To make up for this, you can control all these menus from the keyboard (see below for details).

The video position slider: before you start a video, you can set the starting position (between 0 and 180 minutes, in steps of half a minute). In extended mode you can also set the video position at run time but only if seeking in the file or stream is possible and a duration is known. This will be available a few seconds after starting the video, and it will also be limited to the real duration of the video.

c) Keyboard control

omxplayerGUI (and also omxaudioplayer) supports all keyboard commands of omxplayer with two exceptions:

The Up and Down arrow keys are used for playlist control (go to next or previous song or video in the playlist) instead for large jumps (600 seconds) forward or backward. You have to use PageUp and PageDown instead (or “,” and “.”). See the omxplayer documentation for details or run `omxplayer -k`

from a terminal for a full list of possible keyboard commands.

For some commands omxplayerGUI supports additional options:

Space bar, 'p', Return and Enter on the numeric pad all play/pause a video

'+' and '-' can also be used from the numeric keypad for sound volume control

omxplayerGUI supports its own special commands set using the left ALT key in combination with the following keys:

c - close player (also in omxaudioplayer)

m - toggle Lines: mode between max and min (or toggle between maximize and un-maximize in audio player mode).

f - toggle Lines: mode between full and max (or toggle between full screen and window in audio player mode).

0 ... 9: select Lines: mode between "min" (0) and largest number. Depending on the number of available options, the larger numbers all may have the same effect.

'+' and '-' move through screen ratios forward or backward (between 'auto' and '2.39:1')

u - toggle between simple and extended mode (visible in title bar where an '(f)' is added for simple mode).

s - save playlist as m3u file (also in omxaudioplayer).

h - hide / show all control elements at the bottom

k - the kill switch: sometimes omxplayer hangs and does not return. This command sends a killall command to all omxplayer.bin instances and also to all send-dbus instances.

a - alternate between getting auto mode in background or not. If enabled, an "(a)" will be visible in the window's title bar. (See also get_DAR option in settings description). Only possible in extended mode and when no video is playing.

NumPad+ and **NumPad-** or **PageUp** and **PageDown** - Set the video layer (default and minimum = 0) to a higher or lower value; this will allow overlays. Layer numbers > 0 will be shown in the title bar. Only possible when no video is playing.

If you stop playing a video file, the omxplayerGUI window will switch to playlist view. Handling the playlist is done exactly the same way as in omxaudioplayer (see above).

d) Windowless Mode

omxplayerGUI can also play videos in a windowless mode (depending on special settings, see below). In this case a terminal (xterm) will be started, to blank and fill the whole screen and give keyboard control and then play videos full screen with omxplayer. This is the old mode used by Minimal Kiosk Browser up to version 1.4.

There's a preset installed for this method named "nogui".

7) omxplayerGUI settings

omxplayerGUI can be fine tuned in many ways either by editing its global variables in a special python file (/usr/local/bin/kwebhelper_settings.py) or – much easier - from a web interface in Minimal Kiosk Browser (kweb). OmxplayerGUI shares its global settings with those of kwebhelper.py, another helper file of kweb, responsible for PDF support, downloads and kweb's command interface. When editing the properties of omxplayerGUI you can simply ignore these additional settings.

The omxplayerGUI front end can open this settings page by calling kweb with a special configuration (as root).

You can also call this page from inside kweb, either from its menu page, the control panel or by entering “:s” in the URL entry line. See the kweb manual for details.

a) Choosing presets

On top of the web page, you'll see a few buttons with “presets”, which can be chosen with a mouse click. After selecting a preset, you have to reload the web page to see, which options have been set. Two presets are predefined: “default” (default settings) and “nogui” (windowless mode). You'll see later, how you can create your own presets.



b) List of all options used by omxplayerGUI

The following list will show all settings for omxplayerGUI, in the same way as they are shown in the web interface. Each option can be edited and saved separately.

GENERAL OMXPLAYER AUDIO VIDEO OPTIONS

Options for omxplayer to be used when playing video

omxoptions: Enter one element per line!

 	save
--------------	-------------

Options for omxplayer to be used when playing audio

omxaudiooptions: Enter one element per line!

 	save
--------------	-------------

Special options for watching live tv streams (omxplayer)

omx_livetv_options: Enter one element per line!

--live

save

Add the start of your live tv stream links to this list to enable live tv options

live_tv: Enter one element per line!

save

Mimetypes: if given, this will restrict what omxplayer will be given to play.

mimetypes: Enter one element per line!

save

If omxplayerGUI is not used, omxplayer is started from a terminal (xterm),
to clear the screen and get full keyboard control.

Set the following to "False" to use omxplayer for video without starting a terminal first
(if omxplayerGUI is not used)

omxplayer_in_terminal_for_video: ☒True ☐False

save

Set the following to "False" to use omxplayer for audio without starting a terminal first
(if omxaudioplayer is not used)

omxplayer_in_terminal_for_audio: ☒True ☐False

save

The following list will be used, to detect audio files, especially in m3u playlists

audioextensions: Enter one element per line!

mp3
aac
flac
wav
wma
cda
ogg
ogm
ac3

save

How unknown streams should be handled, must be either 'video' or 'audio'

streammode: video

save

If streammode is set to "video", the following list will be used for checking for video files

videoextensions: Enter one element per line!

asf
avi
mpg
mp4
mpeg
m2v
m1v
vob
divx

save

If the following is set to "True", vlc will be used to play audio files and playlists (audio only)

useVLC: ☐ True ☒ False

save

OMXPLAYERGUI AUDIO & VIDEO OPTIONS

Play audio files or playlists that contain only audio files in omxaudioplayer

useAudioplayer: ☒ True ☐ False

save

Use GUI for playing videos

useVideoplayer: ☒ True ☐ False

save

Volume setting when starting omxplayerGUI ranging from -20 to 4 (-60 to +12 db)

defaultaudiovolume:

0

save

Start playing the first (or only) file automatically

autoplay: ☒ True ☐ False

save

Close the GUI if the last (or only) file has been played to the end

autofinish: ☒ True ☐ False

save

Interface settings for omxaudioplayer and omxplayerGUI (video)

The font to be used for playlist and buttons

fontname:

SansSerif

save

Font size between 10 and 22, will also determine the size of the GUI window:

fontheight:

12

save

Number of entries displayed in playlist window, between 5 and 25:

maxlines:

8

save

Width of the window, value between 40 and 80, defines the minimum number of characters of the song name displayed in the song list (usually much more are shown!), not used for video mode

lwidth: **save**

Minimal height of video area (also depends on fontheight!), 288 or more:

videoheight: **save**

Default 'Lines:' mode, must be one of those: 'min','max', 'full'

screenmode: **save**

Default video mode: set this to 'full' or 'refresh' for full screen, to 'auto' (for automatic detection of the aspect ration) or to one of those: '4:3','16:9','16:10','2.21:1','2.35:1','2.39:1' to play in window (you can also add one additional value here):

videomode: **save**

Set the following to "True" for simple mode (no window resizing, moving etc. while playing video); must be set to True for older omxplayer versions

freeze_window: ☐ True ☒ False **save**

Get aspect ratio in background, if True (if videomode not one of 'auto', 'full' or 'refresh'), costs some processing power and even may block or crash the system, especially with large AVI files, therefore disabled by default. Use it with care.

get_DAR: ☐ True ☒ False **save**

If the following is set to 'True', all control elements are hidden (can be enabled later on with ALT+h)

hide_controls: ☐ True ☒ False **save**

ONLINE VIDEO OPTIONS

Options for pages containing video, either HTML5 video tags or all websites supported by youtube-dl.

If html5 video tags include more than one source format, select the preferred one here

preferred_html5_video_format: **save**

Choose, if HTML5 URL extraction is tried first and youtube-dl extraction afterwards or vice versa

html5_first: ☒ True ☐ False **save**

Additional youtube-dl options, e. g. selecting a resolution or file format

youtube_dl_options: Enter one element per line!

save

Special omxplayer options for web video

youtube_omxoptions: Enter one element per line!

save

c) Creating presets

After editing a number of options you can save the complete actual settings as a preset, that can later be loaded with a simple mouse click.

Save and Load Presets
You may save the current settings as a named preset and load them again later on.
After loading a preset, you should reload this page.

Save Preset

 Name:

Load Preset:

default

guenni

nogui

Delete Preset

 Name:

To save a preset, enter a name and click “Save Preset”. To overwrite it again later with different settings, simply use the same name. To delete a preset, enter its name and click “Delete Preset”.

To load a preset, simply click on the button with its name. Reload the page afterwards to see the active settings. This function is also available at the top of the settings page.

d) Editing the settings directly

You can also edit the settings file directly, if you know, what you are doing. Open it with:

```
sudo idle /usr/local/bin/kwebhelper_settings.py
```

(you can also use a text editor of your choice).

All settings are Python variables of different kinds. Doing something wrong might even crash the program. So you should only do this if you know a bit about Python syntax.

Appendix A: Keyboard commands

Special OmxplayerGUI Keyboard Controls

Key	Mode	Action
ALT c, q	any	quit player
ALT k	playing	kill omxplayer instance, when blocking
ALT m	any	toggle window between max and min
ALT f	any	toggle window between full and max
ALT 0	any	window min size
ALT 1...9	any	other window sizes
ALT s	playlist	save playlist
ALT u	playlist	toggle between simple (f) and extended mode
ALT h	any	hide / show controls
ALT a	playlist	enable / disable auto ratio detection (a)
ALT KP+, PgUp	playlist	layer number + 1
ALT KP-, PgDown	playlist	layer number - 1
ALT +	any	next Mode: (aspect ratio)
ALT -	any	previous Mode: (aspect ratio)

omxplayer(GUI) Keyboard Controls (Play Mode)

OmxplayerGUI uses almost the same keyboards commands as omxplayer

omxplayer	GUI	Action
1	1	decrease speed
2	2	increase speed
<	<	rewind
>	>	fast forward
z	z	show info
j	j	previous audio stream
k	k	next audio stream
i	i	previous chapter
o	o	next chapter
n	n	previous subtitle stream
m	m	next subtitle stream
s	s	toggle subtitles
d	d	decrease subtitle delay (- 250 ms)
f	f	increase subtitle delay (+ 250 ms)
q	q, Esc	exit omxplayer
p, space	p, space, Return, Enter	pause/resume
-	-, KP-	decrease volume
+, =	+, KP+	increase volume
left arrow	left arrow	seek -30 seconds
right arrow	right arrow	seek +30 seconds
down arrow	PgDown, “,”	seek -600 seconds
up arrow	PgUp, .	seek +600 seconds
	up arrow	previous playlist item
	down arrow	next playlist item

Appendix B: Known problems

Extended mode may in some rare cases have problems with video files served via http (see issue <https://github.com/popcornmix/omxplayer/issues/183>), when moving, resizing or hiding the window.

If started with hidden controls and the controls are enabled later on, switching between video aspects may not work, if the window is manually resized to the smallest possible size (which is too small in this case). Select the 'min' value from the “[Lines:](#)” menu in this case.

If you toggle between hiding and un-hiding controls in full screen or maximized ([Lines:](#)) modes, the video is not resized correctly, if the controls are hidden. Hiding them, before calling max or full mode works as expected.

Using the 'get_DAR' option (getting video ratio in background) may sometimes block the player, especially with large AVI files.

All versions of omxplayer later then June 24, 2014 have a problem with live TV streams: if you move or resize a window of omxplayerGUI, the video will not appear any more. You have to stop and restart the video in this case. If you use omxplayerGUI for watching live TV, it's better to stay with version June 24, 2014 or older. You can always get the most recent and all older versions from <http://omxplayer.sconde.net/>.

Appendix C: How to Install a Working Version of Youtube-dl

Youtube-dl is a great tool to extract videos from more than 100 video websites that try to hide them, especially by using some kind of flash player. It's main use is to download videos from youtube (hence the name) and many other websites., but it is also used by many programs (and that includes kweb and omxplayerGUI) to extract the video URL(s) so that the video can be accessed and played directly (no download required). The video websites and the developers of youtube-dl are always in a race condition. If something on a website is changing that may require a change in the (Python) code of youtube-dl. And the developers are also adding new websites to their code all the time.

For this reason youtube-dl has a built-in update function that should be run from time to time:

```
sudo youtube-dl -U
```

Unfortunately this kind of updating youtube-dl doesn't work any more on many linux distributions including new Ubuntu releases and the current Raspbian. And without this update function youtube-dl becomes almost useless within a short time.

If you have had installed youtube-dl some time ago and the update function is working for you, you're lucky and may leave it as it is (but the last part of this tutorial may still be of interest for you). But on recent images or new installations it doesn't work any more and you have to choose one of the following installation methods.

1) This is simple and will get you a working version including the built-in update function. Run the following commands in a terminal:

```
sudo wget https://yt-dl.org/latest/youtube-dl -O /usr/bin/youtube-dl
```

```
sudo chmod a+x /usr/bin/youtube-dl
```

and don't forget to update it quite often with:

```
sudo youtube-dl -U
```

This solution will work, but it has one big disadvantage: It will be rather slow. Due to the way youtube-dl is built, it will take about 12 seconds to extract a video address on a Raspberry Pi running in turbo mode (and much longer, if your Raspberry Pi is not overclocked). So you might consider solution 2.

2) The second method is a bit more complicated but will install a version that runs almost twice as fast (and if you access a lot of web videos that will mount up to hours of waiting time you can spare). To make life easier for you, the kweb distribution now includes two scripts, that will take care of everything. From a terminal run

```
ginstall-ytdl
```

This will remove older versions of youtube-dl and install the most recent version from github instead. After the installation process you'll find a folder "youtube-dl" in your home directory, which you shouldn't remove (this would break your installation). If git is not installed in your system it will also be installed and you shouldn't remove it afterwards because it is also needed for updating youtube-dl. This is done by another script:

```
update-ytdl
```

You should run this as often as possible. The github version is changing almost every day.