

VirtualDubMod is a project that was born when suddenly a lot of modifications to the original VirtualDub by Avery Lee sprung up, mainly on the Doom9 forums. Some people got tired of needing several diffrent modified VirtualDub executables, so the idea arose to put them all together into a single app, VirtualDubMod.

The founders were:

- VirtualDubOGM Cyrius
- VirtualDubMPeg2 Pulco-Citron
- VirtualDubAVS Belgabor

Extended features in VirtualDubMod 1.5.x

Nb : VirtualDubMod now has a basic Unicode support. However this support is only available in Windows versions that handle it natively (2k/XP handle it). Win9x doesn't handle Unicode natively and so you won't have Unicode support in VirtualDubMod with this OS.

This short documentation give you some information on the following subjects in VirtualDubMod 1.5.x :

- Preferences
- AVI Information
- Image formats
- Streams menu
- Multiple streams handling
- Comments and chapters handling
- Previewing and Saving
- Batch mode
- Miscellanea

Preferences

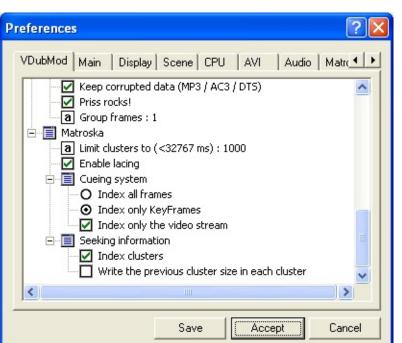
Since version 1.5.4.x VirtualDubMod also uses an extended tree list thanks to which you can access all its preferences.

Depending on which kind of preference it is, clicking on an item in the tree will make appear a combo box or an edit box (optionally an icon to popup a file selector).

For enabling / disabling preferences (checkbox / radio button) click on the checkbox or radio button.

The default video mode (*Direct stream copy, Fast recompress, Normal recompress, Full processing mode*) in which VirtualDubMod starts can be changed in the *Main* preferences.

The *Display* preferences let you choose to enable/disable by default the *Display input video* and *Display output video* of the *Options* menu. You can ask for the window to be automatically resized (according to the size of the opened video) and centered. You can also specify a specific language dll to use (all or part of the strings / messages in the program will be affected).



Two new AVI options have

been added in the AVI Preferences.

The first one let you specify the default behaviour of VirtualDubMod concerning MP3 VBR ala Nandub in AVI files. You can either force VirtualDubMod to handle the stream the same way Nandub does, or let the program fix the VBR stream as VirtualDub does.

Note that you will be asked to confirm this behaviour (or to use the other one) if a VBR stream is detected when opening an AVI file (just click *Yes* if you want to use the behaviour you specified in the preferences).

The second one concerns the way to handle AC3 and DTS sources when saving to AVI. It has been reported many times that Nandub (and VirtualDub) cut an AC3 track anywhere (and generally in the middle of an audio frame) when editing an AVI file. This sometimes lead to troubles when using the AC3 track with other tools.

You can ask VirtualDubMod to write the size of the audio frames (only possible with CBR streams) in the AVI headers. The specs theorically allow it but actually programs generally fix this value to 1 (meaning that the program can cut the stream on every byte if it wants).

Writing the size of the frames in the header should force other editing tools to deal with full audio frames (and so it shouldn't cut anywhere). Use this feature AT YOUR OWN RISKS!!

A new *Audio* section have been added in the *Preferences*. This tab let you access 5 options in VirtualDubMod :

- **Default preload** : default preload value (in ms) to use for audio streams
- **Disable preload for non AVI output** : let you disable preload when not saving to AVI
- **Priss rocks!** : let you use *Priss* (VirtualDub internal Mpeg audio decoder) instead of an ACM one when decoding MP3 audio. There may be some bugs (in the way it is implemented, not in Priss ;)) but it seems to work better than ACM on MP3 VBR :)
- **Group frames** : minimum number of audio frames to pack in a block. 0 means to process one frame at a time. Only effective when not saving to AVI, not processing MP3 VBR ala Nandub, or when using the *Frame mode* (see before)
- **Keep corrupted data** : tells VirtualDubMod what to do with corrupted data found in MP3 / AC3 / DTS files. When kept the program will try to keep the stream in synch by processing the corrupted data as real data (computing the equivalent number of samples corresponding to the size of the corrupted data)

And a *Matroska* section has also been added for those who want to test different settings.

AVI Information

The field *Preload skew* in the *AVI information* window give you the number of blocks and an estimated number of samples used for preloading. Two fields have been added in this window :

- **FourCC code** : give you the FourCC code of the compression format for the video. Note that the *Decompressor* field doesn't necessarily give you the codec in which the video has been compressed (e.g. DivX 3/4 video is decompressed by the DivX 5 decoder). The second line give you the name (if known) of the format corresponding to this FourCC code.
- Audio tag : give you the compression format (the wFormatTag value) of the audio stream. Tells you if an ACM decoder is available for this format (the field *Compression* give you the name of the ACM codec in this case). The second line give you the name (if known) of the format corresponding to this tag.

Note that the name of the AVI Information format is retrieved from Video stream the Codecs.ini file (that Frame size, fps (µs per frame) 640x480, 23.976 fps (41708 µs) should be in your Length: 70992 frames (49:20.96) FourCC : [XVID] VirtualDubMod FourCC code: XviD MPEG-4 Codec directory). This file Decompressor: XviD MPEG-4 Codec Number of key frames: 850 comes from The Core Min/avg/max/total key frame size: 3333/38776/82471 (32188K) Media Player. We thank Min/avg/max/total delta frame size: 163/6438/79706 (441056K) Data rate: 1309 kbps (0.35% overhead) its developpers for this great player, having Audio stream made this file and letting Sampling rate: 48000Hz Channels: 2 (Stereo) us use it :). Sample precision: N/A Tag : 0x0055, ACM Decodable Audio tag: MPEG Layer 3 Compression: Fraunhofer IIS MPEG Layer-3 Codec Layout: 70981 chunks (24000|0.50s preload) Length: 47374945 samples (49:20.93) Min/avg/max/total frame size: 244/667/8000 (46265K) Data rate: 128 kbps (3.47% overhead)

OK

Image formats

VirtualDubMod now use the **Corona** library. This library handle BMP, TARGA, PNG, JPEG, GIF and PCX image formats on input, and PNG on output.

This means that you can open image sequences in one of those formats (BMP and TARGA are still handled by VirtualDub internal routines though), and save to a BMP (using VirtualDub BMP routines), TARGA (using VirtualDub TARGA routines) or PNG (using Corona PNG routines) sequence of images.

The default selected format is TARGA. When using multiple times this feature during a same VirtualDubMod session the last used format is kept in memory and selected on next calls.

Filename prefix:	The Irresponsible Captain Tylor - Ep20 -
Filename suffix, including extension:	.png
Minimum number of digits in name:	5
Directory to hold images:	
First frame filename:	The Irresponsible Captain Tylor - Ep20 - 00000.png
Last frame filename:	The Irresponsible Captain Tylor - Ep20 - 33712.png
Last frame filename: Output format	PNG
	PNG Windows BMP TARGA

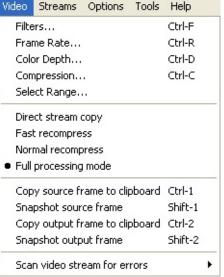
A new feature has been added in the *Video* menu. This feature let you save the current input or output image to a BMP (using VirtualDub BMP routines), TARGA (using VirtualDub TARGA routines) or PNG (using Corona PNG routines) file. This feature is available thanks to the new *Snapshot source frame* and *Snapshot output frame* items.

The default selected format is TARGA. When using multiple times this feature during a same VirtualDubMod session the last used format is kept in memory and selected on next calls. This default

selected format is shared with the *Save image sequence* feature (i.e. if you saved an image in PNG format the default format selected when saving a sequence of images will be PNG, and vice-versa).

Streams menu

Starting from version 1.5.x VirtualDubMod allow the use of multiple streams in its standard use (e.g. when writing AVI files).

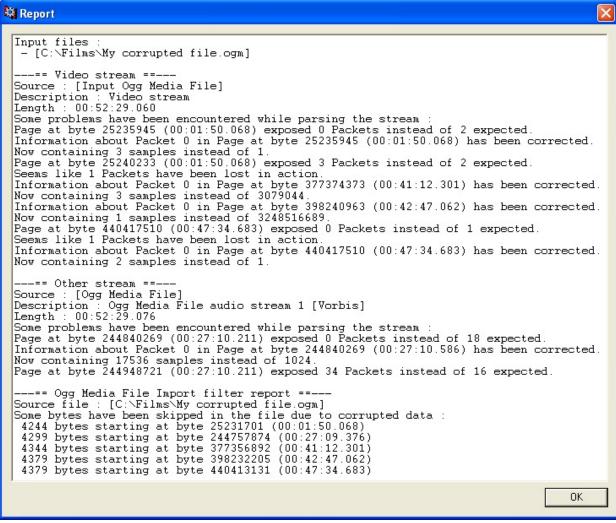




In this purpose the *Audio* menu has been replaced by *Streams* in the main menu. This new menu let you access features for the Video stream and the list of currently available streams (*Stream list*).

The *Report* item let you access a window where a report on currently opened files and available streams appears.

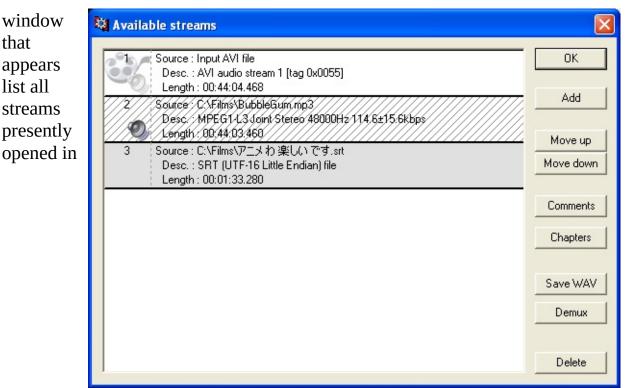
This reports give you general and specific information for each opened file or stream (caracteristics, errors, ...), such as this one :



P.S. : if your report looks like this one then you are really unlucky :p

Multiple streams handling

You can manage the available streams thanks to *Streams -> Stream list*. The



VirtualDubMod (except the video stream of course).

Pictures and colors give you some more information on a stream. The currently selected stream has a grey (instead of white) backgound in the list. If a stream is disabled (i.e. you don't want it when muxing) some hatchings are drawn on it. You can disable a stream by double-clicking on it.

means the stream comes from the currently opened file in VirtualDubMod. An audio streams with is selected for previewing while one with isn't (just click on this image to change the state of an audio stream regarding preview). Note that only one audio stream can be selected for previewing (you can also select none), and that you can select a disabled stream. Since version 1.5.x VirtualDubMod includes the **liba52** library. Thanks to this library you should be able to decompress (i.e. use in *preview* or in *Full processing mode*) an AC3 track (the stream is converted to *Dolby surround compatible stereo* by the library). Note that an option also allow you to use *Priss* (internal Mpeg audio decoder) instead of an ACM one to decode an MP3 track (see before).

You can change the order of the streams by using *Move up* or *Move down*. You

can add external sources with the *Add* button, or remove a stream from the list with the *Delete* button (note that streams coming from the opened file can't be removed from the list, to select/deselect such a stream you need to disable it thanks to the *(Un)Select* button - the *Delete* button become *Select* or *Unselect* for this kind of stream - or by double-clicking on the correspondant item in the list).

Currently supported external sources are :

- WAV files
- **MP3** files
- AC3 files
- **Ogg Vorbis** files
- **DTS** files
- **SRT** files (ASCII/Unicode)

Note that for MP3 files the program look if there is a RIFF header at the beginning (i.e. it is a WAV file but you forced the program to open it as MP3) or tags (at the beginning or end of the file). Those data (RIFF header and tags) aren't muxed in the output file.

For AC3 and DTS files the program also check if there is a RIFF header at the beginning of the file.

In any case you will find all those information (was there a RIFF header, tags, and or data seen as corrupted in the file) in the *Report* window.

As you can see Unicode filenames are also supported (in Windows version that support it).

When opening an external source a dialog box showing progress of the process will appear. If the file wasn't valid it won't be added in the stream list. You can verify all went well thanks to the *Descripion* or *Length* information given in the stream list

MPA File Import Filter	
Parsing File	
25480K of 36974K	Abort

(if it is far different from what you expected maybe something went wrong). The external sources are parsed and all necessary information are stored in memory for faster data access afterwards. The amount of memory used depends of the type of file opened, and its content. Some improvements have been made since the previous version of VirtualDubMod ;).For standard (i.e. CBR) AC3/DTS files it should be a few hundred bytes, around 1MB for an MP3 source (1 hours) and 2MB for an Ogg Vorbis file with same duration. Subtitles are completly stored in memory but this doesn't represent an excessive amount

of memory :p

To access audio streams features you can right-click on the correspondant item in the list. The *Audio* menu will then popup. Note that the *mode* (i.e. *None, AVI audio, WAV audio*) have been removed from the menu since useless now.

	Interleaving
	Compression
	Use advanced filtering
	Filters
	Conversion
	Volume
•	Direct stream copy
	Full processing mode

You can demux a stream with or without a WAV header by selecting *Save WAV* or *Demux*.

Ogg Vorbis streams will be automatically demuxed to an Ogg file.

Text streams	Demux SRT stream	n			? 🛛
will be automatically	Enregistrer dans :	C Temp	▼ 🗢 Ē	* 📰 •	
demuxed to a					
subtitle (SRT)					
file.					
You will have to					
choose in which					
format (ASCII,					
UTF-8 or	Nom du fichier :			[Francistan
Unicode) you	Nom du tichier .			l	Enregistrer
want to save the	Туре:	SRT file (*.srt)		-	Annuler
subtitles.	Save subtitles as :				
The default		ASCII UTF-8 Unicode			

selected format

depends on the format of the subtitles you want to save. If subtitles are ASCII then the default proposed format is ASCII, otherwise it is Unicode.

Comments and chapters handling

You can access specific comments or chapters for a stream thanks to *Comments* and *Chapters*. The video comments and chapters are accessible thanks to *Video comments* and *Video chapters* in the *Streams* menu.

Generally you put your chapters in the video stream. Note that only the comments and chapters specified in the video stream are taken into account

when saving to a file that only handle comments or chapters at the file level (and not at the stream level). This is for example the case for AVI files (which only handle comments). And comments or chapters coming from such files will only be added in the video stream.

😫 Stream	n comments	
1	Type : Specific (Tag : [ISFT]) Value : [Nandub v1.0rc2]	OK
2	Type : Title (Tag : [INAM]) Value : [Bubblegum Crisis - Episode 5 of 8]	Cancel
3	Type : Subject (Tag : [ISBJ]) Value : [Bubblegum Crisis - Episode 5 of 8]	Move up
4	Type : Author (Tag : [IART]) Value : []	Move down
5	Type : Copyright (Tag : [ICOP]) Value : []	
6	Type : Comment (Tag : [ICMT]) Value : []	Delete
7	Type : Comment (Tag : []) Value : [スイリクわ 阿呆 です]	Add
8	Type : Comment (Tag : []) Value : [エイバリーわ 神様 です]	Edit
Generic	▼ Tag:	
	Value :	

Comments have 3 properties :

- *Type* : defines the type of comment (see below)
- *Tag* : the name of the comment (only matters for *Generic* comments)
- *Value* : the value of the comment

There are various type of comments. Types are a convenient way to use the same comment for any output format (the output functions automatically format a comment according to its type). Note that some output formats doesn't handle comments, or only some types (not all). Here is the list of currently defined comments under VirtualDubMod 1.5.x:

Туре	Decription	Handled for AVI	for	Handled for MKV
	Comment specific to a container. This kind of comment can only come from a file (you can't			

Specific	insert one yourself) and won't be included in the output file unless both formats (input and output) are the same or you edit its type yourself.	*	*	No
Generic	Generic comment (use this type of comment and fill the <i>Tag</i> field if none of the type used suits you).	Yes	Yes	Yes
Software	Software used to make the file. Note that you can't edit this		No	Yes
Title	Title of the clip/track (Title property in WMPlayer).	Yes	Yes	Yes
Description	Subject of the clip/track		Yes	Yes
Language	anguage Language property in WMPlayer.		Yes	Yes
Author	The artist generally considered responsible for the work. In popular music this is usually the performing band or singer. For		Yes	Yes
Copyright	Copyright attribution (Copyright property in WMPlayer).	Yes	Yes	Yes
License	License information, eg, 'All Rights Reserved', 'Any Use Permitted', a URL to a license, etc.	No	Yes	Yes

Person or organisation that Commissioner commissioned the subject of the file.		Yes	No	Yes
Comment	Comment on the clip/track.	Yes	No	Yes
Genre		Yes	Yes	Yes
Keywords	Keywords refering to the file or its subject.	Yes	No	Yes
Date	Creation date.	No	Yes	No
Version	The version field may be used to differentiate multiple versions of the same track title in a single collection (e.g. remix info).		Yes	No
PerformerThe artist(s) who performed the work. In classical music this would be the conductor, orchestra, soloists. In an audio book it would be the actor who did the reading. In popular music this is typically the same as the Author and is omitted.		No	Yes	Yes
Album	The collection name to which this track belongs.	No	Yes	Yes
Track Number	Track The track number of this piece if		Yes	No
Organisation Name of the organization producing the track (i.e. the 'record label').		Yes	Yes	No
Location Location where track was recorded.		Yes	Yes	No
Contact	Contact information for the creators or distributors of the track. This could be a URL, an email address, the physical address of the producing label.	No	Yes	No

ISRC	ISRC number for the track.	No	Yes	Yes
URL		No	No	No
(Track) Name	Name of the track (as described in AVI or Matroska specs).	Yes	No	Yes
As you could see Unicode comments are				

As you could see Unicode comments are supported here too (except for Win9x as said at the beginning).

	110	110					
	Yes	No	Yes				
Γ	Language Tag:						
J	Japanese Value : Ja						
🔲 Full language list							

Nb : please note that when saving to AVI

comments are translated to the ANSI code page (in my tests I wasn't able to properly include Unicode comments in an AVI file).

When selecting (or modifying) a *Language* comment another combo-box containing a list of predefined languages will appear. This list languages defined in the **ISO-639-1** standard. You can access the full list of languages defined in the **ISO-639-2** standard by checking the *Full language list* box.

Chapters are now handled in another window. Time is stored as fraction in the code, but in the GUI the format used is *HH:MM:SS.mmm* (*HH* = Hour; *MM* : Minute; *SS* : Second; *mmm* : Millisecond). If the time you entered in the field doesn't follow this format you will be warned when trying to *Add* or *Edit* a chapter information.

You can use Unicode chapter names too (unless under Win9x).

You can also import a chapter list (.txt file) thanks to the *Import* button. Known files formats are:

- SmartRipper
- chapterXtractor (select OGG format)
- **DVDDecrypter** (select *OGG format*)

🔯 Stream chapters	
1 00:00:03.000 Name : [Ангелы и демоны кружили надо мной]	ОК
2 00:00:08.000 Name : [Рассекали терни и млечные пути]	Cancel
3 00:00:14.000 Name : [Не зная о чем сделал он покой]	
4 00:00:20.000 Name : [Кто его зова понять не смог]	Import
5 00:00:26.000 Name : [Mala dio fasis]	Delete
	Add
	Edit
Time : 00:00:00.000	
Name :	

Previewing and Saving

A few things have been changed in the *File menu*.

Since you can now append OGM segments, *Append AVI segment* has been renamed *Append segment*. When using this feature the program will select the correct extension of files to append (i.e. AVI files when you have opened an AVI file, and OGM files when you have opened an OGM file).

File	Edit	Video	Streams	Options	Tools		
0	Open video file Ctrl-O						
0	Open video file via AviSynth Ctrl-V						
A	ppend	segmen	it				
Preview Ctrl-F5					F5		
Preview VBR Shift-F5					:-F5		
Save As F7							
Close video file Ctrl-W							

Comments and chapters should be appended correctly too. If comments are equals duplicated ones are removed. If chapters seems equals (same name, and time difference between them is less than 2 ms) then the duplicated ones are removed.

A new *Preview VBR* item has been added. Using this preview mode (instead of the normal *Preview*) should give better results when the audio stream you

selected for preview is VBR (especially when you start previewing in the middle of the clip and not at the beginning).

The previous *Save as AVI*, *Save old format AVI*, *Save segmented AVI* and *Save as OGM* items have been replaced by a unique *Save as* window.

The new <i>Save as</i>	Save As				? 🔀
item let you	Enregistrer dans : 🧯	🗅 Onegai Twins 🔹	- 🕁 🖻) 💣 🎟 -	
access a window		We might be strangers.mkv			
you can use to					
save in the					
desired format,					
with the desired					
options.					
This window	Nom du fichier :				Enregistrer
merge the	, T				
options so that	,	Matroska File (*.mkv)			Annuler
you can save in	Don't run this job not	Audio-Video Interleave (*.avi) Ogg Media File (*.ogm)			
the old AVI	🔲 Save AVI in old 1.0 (Matroska File (*.mkv) VIW 1.1ej format (Compatibility moda	ej		
format (AVI 1.0,	Segment output file				
VfW 1.1e; the	Limit number of	video frames per segment to	100	frames	
default format	Limit size (in Megal	Bytes) of each segment to :	2000	MB (50-2048)	
being AVI 2.0,	Video				
OpenDML	Video mode : Direc	t stream copy	•		
1.02).	Compression : (No re	ecompression: DIVX)		Change	
being AVI 2.0, OpenDML	Video mode : Direc	••	•	Change	

You can select

the desired output format (i.e. AVI, OGM or MKV file).

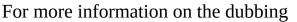
The format used in the last call of *Save as* is stored in the Windows registry and automatically selected each time you access the window.

You can also ask to segment the output file (according to a size or number of frames limit).

Note that suring a session of VirtualDubMod the program remembers if you used the Job list to queue the previous job or not. i.e. if you put a job in the list, next time you call *Save as* then the corresponding checkbox will be checked by default.

This window also remind you of the current video mode, and the selected codec (when reencoding). You can change the mode and/or change the codec to use for reencoding.

The *Main* tab in the *Dubbing status* window remains unchanged : it still show you status about the *video* stream and an *audio* stream. The *audio* section actually give you information on the first stream being muxed in the output file (i.e. it may in fact be a text stream). The *Estimated total time* is also computed only by taking into account the *video* and *audio* status. However, even if no information is shown about other streams (if any) the *Projected file size* and the *Progress* information take into account all the streams being muxed.



VirtualDubMod Status - [Hack Si... Streams Video Perf Main Video frame: 15753/39973 🛆 Stream 1 sample: 10520513/26675102 Stream 2 sample: 27395/69467 Video data: 89.7MB (140KB/s) Stream 1 data: 10643KB Stream 2 data: 10026KB Projected file size: 278.82MB Rendering rate: 1166.0 fps Time elapsed: 0:17 Progress: Processing thread priority: Normal -Show input video Abort Show output video

process a new *Streams* tab have been added. This tab still give you information about the *video* stream, but also list information about all the other streams (in the order in which they are muxed in the file). The number given for each stream is not an index but the serial given to the stream in the output file. This means that each number is unique for a dubbing process, but (even if it is most likely to be like this) they are not necessarily 1, 2, 3, ... (e.g. having 7803, 204, 65214 doesn't always mean there is a bug :p)

Here both *Projected file size* and *Estimated total time* take into account all the streams.

As you can see a scrollbar appears in the tab when the number of streams make it necessary (i.e. when you have more than 1 streams - not including video - to mux).

All the external files you added as input (and all the video comments and chapters) are removed from the list when closing the currently opened video file. This happens when you open a new video file, when you select *File -> Close video file* in the menu, at the end of a job, and (of course) when leaving VirtualDubMod.

So don't open those external files before opening your video file because in this case you will have to do it again :p

Batch mode

Last but not least the Job file will save all the information concerning used streams and their options (compression, ...), comments and chapters. So you can still use the *Don't run this job now; add it to job control so I can run it in batch mode* feature.

Start
Abort
Skip

Note that a *Skip* button has been added in the *Job list* window. On the contrary to the *Abort* button (which stop jobs processing) it abort the current job and go the next one.

Also your current configuration (input streams selected and their associated comments and chapters) should be saved when using *Save processing settings*, and can be reloaded when using *Load processing settings*.

Miscellanea

The *Go to last keyframe* feature (coming from Nandub) has been added in the *Edit* menu. This feature let you jump to the nearest previous keyframe preceding a certain position (expressed in megabytes) in the file.

You can see the position in file of the currently shown frame at the bottom of the window.

The file position only take into account the currently opened video file, and so is only worthy when you plan to save (in *Direct stream copy*



mode) to the same file format (AVI and OGM don't have similar overheads).

Scripting with VirtualDubMod

Like VirtualDub, VirtualDubMod uses Sylia (developped by Avery Lee too). Some specific functions have been added though.

You will find below a list of the functions you can use in a script with VirtualDubMod.

This list also give you the type of parameters you can use with the function. There are 2 possible types :

- : signed long integer (32 bits) which value can be from -2147483648 to +2147483647
- : a standard string

Integers can also be used as boolean values : 0 means false and any other value means true.

In the functions described below the values returned (when any) are integer values.

You will see that some parameters use the first letter of their name to describe their type :

- b : boolean value
- f : this value is a flag (generally a boolean) in VirtualDub settings (e.g. 'Preferences' settings)
- n/l : generally a long integer (sometimes it is used as DWORD, i.e. unsigned long integer)
- w : generally means the value is a WORD (unsigned integer)

Since version 1.5.5 VirtualDub support Unicode filenames. Thus filenames used in the script files are actually converted to UTF-8, and then processed by the strCify function to make them fit in the script as a string. Here is the list of defined functions :

- General functions (VirtualDub.):
- Subset functions (VirtualDub.subset.):
- Video functions (VirtualDub.video.):
- Video filters functions (VirtualDub.video.filters.):
- Video filters functions (VirtualDub.video.filters.instance[n].):
- Stream (mainly audio, sometimes text) functions (VirtualDub.stream[n].)

- Audio filters functions (VirtualDub.stream[n].filters):
- Audio filters functions (VirtualDub.stream[n].filters.instance[n]):

For compatibility reasons some functions that are somehow deprecated in VirtualDubMod 1.5.x are still kept.

For the same reasons note that **VirtualDub.audio** will be interpreted as **VirtualDub.stream[0]** for keeping compatibility with VirtualDub scripts : All this should make VirtualDub scripts still be interpreted correctly in VirtualDubMod 1.5.x.

Here is a list of possible comment types (see the documentation for a description of those types) :

Type	Value
Specific	0x00000000
Generic	0x0000001
Software	0x0000002
Title	0x0000003
Description	0x00000004
Author	0x00000005
Copyright	0x0000006
License	0x00000007
Commissioned	0x0000008
Comment	0x0000009
Genre	0x0000000A
Keywords	0x000000B
Date	0x0000000C
Version	0x000000D
Performer	0x000000E
Album	0x000000F
TrackNumber	0x0000010
Organisation	0x00000011
Location	0x00000012

Contact	0x0000013
ISRC	0x0000014
URL	0x0000015
(Track) Name	0x0000016
Language	0x00010000

Note that the value 0x00010000 means a user-defined language. Languages defined in the **ISO-639-1&2** standards use values beyond 0x00010000 ...

Here is a list of possible input types :

Туре	Value	
WAV	0x0201	
MP3	0x0202	
AC3	0x0203	
Ogg Vorbis	0x0204	
DTS	0x0205	
SRT	0x0301	

Here is the list of functions that have been removed :

Here is a list of commandline options you can use when launching VirtualDubMod.

First of all options coming from VirtualDub :

Option	Comment
/s"ScriptFile.vcf"	Execute the specified script file.
/c	Clear the job list.
/p"srcfile","dstfile"	Add file batch process to job list.
/b"srcdir","dstdir"	Add directory batch process to job list.
/r	Run the job list.
/x	Exit when jobs complete.
/h	Disable the crash handler.
/w	Set the global <i>g_fWine</i> flag to true, which make some display function behave differently. Maybe fix some issues when running the program under <i>Wine</i>
/fsck	Make VirtualDub crash (to test the crash handler).

Note that when specifying file or directories to process there are no spaces between the switch and the name specified between quotes.

Now options added in VirtualDubMod :

Option	Comment	
/log	Generates the <i>vdenc.log</i> log file when recompressing video. Each line give information about a frame (first number gives the kind of frame - 1 for keyframe, 0 for others -, the second number give the size of the compressed frame.	
/log"LogFile.log"	og"LogFile.log " With this one you can specify where will be written the log file.	
	This one prevents VirtualDubMod from actually writing the video data in the output file.	

the *log* and *nowrite* options have been proposed and added by *jonny*, author of **DivX5Enc**.

VirtualDubMod FAQ

What do I need to run VirtualDubMod?

Read this if VirtualDubMod doesn't run for some reason!

Get the latest release and unpack it somewhere on your HD. Get the latest Dll pack and unpack it in the same directory. If you don't have all the VirtualDub Dlls (sylia.dll, vdrslink.dll, etc.) get them (either coming from the official VirtualDub, or go get the latest special Dll pack with those files from SourceForge).

If you want AviSynth syntax coloring you also need AviSynthLexer.lexer. Syntax coloring for commands from external plug-ins works since AviSynth 2.5.2.

There will be also some translation packs available.

I found a bug / I want to propose this feature. What should I do?

- 1. All bug reports and feature requests go to the Trackers on our <u>project page</u> <u>on SourceForge</u>. You don't need to register there to be able to report, but doing so has a big advantage. Whenever somebody comments on your request, you will get a notification email, so you will asap know when something is up.
- 2. Make sure your request hasn't been reported before! We (the developers) close the items that are resolved, so make sure you select "All" in the status box on the respective tracker (normally only the open items are shown). Please take also the time to look at the current Changlog. You can find this via the SF CVS browser (CVS->Browse CVS Repository->VirtualDubMod15->VirtualDub->mod->res->Changes Mod.txt (click on the version number right of the file name))
- 3. The status of all bugs regarding 1.4.x has been changed to 'Pending' after the release of 1.5.x. This was done to get rid of stuff that doenst apply anymore due to the major changes. If you find your bug to still exist, please change its status back to open instead of submitting it anew.

Of course you can also post your stuff in the forum for discussion, but **always** add it to the respective tracker, otherwise bookkeeping is hell for us and items might slip out. We all are only humans, so we might overlook important stuff if

it's not collected at a central point.

Where has the audio menu gone to?

Right mouse button in the stream list.

How do I use 'Open via AviSynth'?

This allows you to open any AVISynth compatible video file by automatically generating a suitable script by a selectable template. These templates go into the 'templates' directory in your VirtualDub directory. Template syntax is the same as for AVISynthesizer (I included readme_AVISynthesizer.txt because as of this writing the AVISynthesizer homepage is not available). You can choose the default template shown in the open dialog by starting the template description with "Default" (note that the template filename has nothing to do with this). *This function has no intelligence whatsoever!* You're responsible to choose the right template for the file (eg a SegmentedAVISource template for multi-segment avis or giving the right fps to DirectShowSource) For this to work flawlessly you need AVISynth 2.05 and a correctly set plugin directory. (Create a registry key called "PluginDir" in HKEY_LOCAL_MACHINE\Software\Avisynth\) A workaround if you don't want to do this is adding the needed LoadPlugin lines into the templates.

Why is using vbr mp3 in avis a bad idea?

We discussed this issue at length in irc and Cyrius (suiryc) gave a pretty good explanation:

```
[21:33] <Belgabor|Home> cyrius, what did your experimets tell?
[21:33] <Suiryc> Belgabor | Home : I think I know know why VBR is not
[21:33] <Suiryc> s/know/now
[21:33] <Belgabor|Home> ok, tell me
[21:33] <Suiryc> :)
[21:34] <Suiryc> first of all there are 2 'headers' in the AVI (audi
[21:34] <Belgabor|Home> I have the feeling i need to hammer that dow
[21:34] <ChristianHJW> lol
[21:34] <Suiryc> first one is a general one (the same struture is us
[21:35] <Suiryc> AVISTREAMINFO
[21:35] <Suiryc> (IIRC ... there should use shorter names ...)
[21:35] <Belgabor|Home> lol
[21:37] <spyder482> ChristianHJW: I won't be moving for a few months
[21:37] <Suiryc> this one tell how many frames there are in the stre
[21:37] <Suiryc> and what is the rate of the frames
[21:37] <Suiryc> thanks to dwRate & dwScale fields
[21:37] <Belgabor|Home> got that
```

[21:38] <Suiryc> it also contains a field saying the size of 1 frame [21:38] <Suiryc> if VBR, then it is set to 0, otherwise it is set to [21:38] <Belgabor|Home> dwSampleSize [21:39] <Suiryc> yep [21:39] <Suiryc> then there is a header specific to the audio stream [21:39] <Suiryc> this one tell the samplerate (44100, 48000, ...) [21:39] <Suiryc> the byterate [21:39] <Suiryc> the format (wFormatTag) [21:40] <Suiryc> and especially contains a field names nBlockAlign [21:40] <Suiryc> nBlockAlign tell how many bytes an audio frame cont [21:40] <Suiryc> _BUT_ [21:40] <Belgabor | Home> And that musnt be 0 [21:40] <Suiryc> cannot be set to 0 [21:40] <spyder482> so much work for AVI... [21:40] <Suiryc> :) [21:40] <Belgabor|Home> ok, i think i get the picture [21:40] <Suiryc> ok so let's continue [21:41] <Belgabor|Home> ok [21:41] <ChristianHJW> all with you guys ... [21:41] <Suiryc> in Nandub here is what happens with an MP3 stream ([21:42] <Suiryc> Nando set dwRate to the samplerate (44100, 48000, . [21:42] <spyder482> don't you two have a channel for this? :) [21:42] <Suiryc> spyder482 : shut up :P [21:42] <Suiryc> and set dwScale to 1152 [21:42] <spyder482> lol [21:42] <Belgabor|Home> no, the other one is just for lurking [21:42] <Belgabor|Home> :p [21:42] <Suiryc> :] [21:42] <spyder482> hehe [21:43] <Suiryc> and set nBlockAlign to 1152 too [21:43] <Suiryc> then, when muxing it only treat whole MP3 frames [21:43] <Suiryc> (i.e. each MP3 frame is in its own Chunk) [21:44] <Suiryc> you still follow ? [21:44] <md`> who has done the mpeg2 import part of vdmod? [21:44] <Belgabor|Home> ok, one mp3 frame is what? [21:44] <Belgabor|Home> pulco-citron [21:44] <md`> hmpf [21:44] <spyder482> pulco-citron [21:44] <spyder482> oh [21:44] <spyder482> :) [21:45] <md`> why does he generate d2v and dont let the user decide [21:45] <Belgabor|Home> dunno [21:45] <md`> if there is one already [21:45] <md`> hmmm [21:45] <Suiryc> Belgabor|Home : an Mpeg1-Layer3 frame is the shorte [21:45] <ChristianHJW> let Suiryc finish guys .. please [21:45] <md`> yes ok [21:45] <Belgabor|Home> ok [21:45] <spyder482> ChristianHJW: check #virtualdub [21:45] <Suiryc> it contains an header saying what is in the frame,

[21:46] <ChristianHJW> we have to know whats wrong in AVI to be able [21:46] <Belgabor|Home> this is how much data? [21:46] <Suiryc> somehow 1 MP3 frame ~ 1 video frame [21:46] <Belgabor|Home> ChristianHJW: lol [21:46] <Suiryc> the size of a frame depends on the MP3 settings [21:46] <Suiryc> (i.e. bitrate, ...) [21:46] <Belgabor|Home> ok [21:47] <Belgabor|Home> is it fixed for a file or varible in vbr? [21:47] <Suiryc> however a Mpeg1-layer3 frame conatins 1152 samples [21:47] <Suiryc> the size of a frame is variable [21:47] <Suiryc> even in CBR [21:48] <Suiryc> (e.g. frames will be of 417 or 418 bytes) [21:48] <Belgabor|Home> ok, but 1152 is the upper limit? [21:48] <Suiryc> because a fixed btrate must be achieved [21:48] <Suiryc> 1152 is the number of samples a frame contains [21:48] <Suiryc> each frame (whatever its size may be) contains 1152 [21:49] <Belgabor|Home> oic [21:49] <Suiryc> so let's continue ;) [21:49] <Suiryc> each frame contains 1152 samples [21:49] <Belgabor|Home> ok [21:49] <Suiryc> and the rate of the stream (in AVISTREAMINFO) has b [21:49] <Suiryc> dwRate / dwScale = SampleRate/1152 [21:50] <Suiryc> since each Frame contains 1152 it is equal to the ' [21:50] <Suiryc> (as for video) [21:50] <Belgabor|Home> ok, i think i got that [21:50] <Suiryc> now you must recall that each frame is in its own A [21:50] <Belgabor|Home> ok [21:50] <Suiryc> so it is also the 'chunkrate' [21:51] <Suiryc> so here is now what happens (it is most likely what [21:51] <Belgabor|Home> ic [21:51] <Suiryc> WMP will get both headers [21:52] <Suiryc> which will say to it that the rate of the stream is [21:52] <Belgabor|Home> gimme a sec, brb [21:52] <Suiryc> and that each audio frame is 1152 bytes long (nBloc [21:52] <Suiryc> k [21:53] <Belgabor|Home> back [21:54] <Suiryc> ok so WMP believe each frame is 1152 bytes long [21:54] <Belgabor|Home> yeah [21:54] <Suiryc> which is not the case (generally frames are around [21:55] <Suiryc> but [21:55] <Belgabor|Home> yeah, got that much [21:55] <Suiryc> now you are reading data in the file [21:55] <Suiryc> and WMP needs to know when to read the audio [21:55] <Suiryc> (i.e. to which time correspond an audio frame) [21:56] <Suiryc> to do so it will look at all the previous audio chu [21:56] <Suiryc> for each shunk it divide the size (in bytes) of the [21:56] <Belgabor|Home> ok [21:56] <Suiryc> s/shunk/chunk [21:57] <Belgabor|Home> ok [21:57] <Suiryc> (since every tools dealing with the stream must cut

[21:57] <Suiryc> since each chunk is shorter than 1152 bytes (nBlock [21:57] <Suiryc> but this is not possible [21:58] <Suiryc> since tools work on blocks of nBlockAlign bytes, it [21:58] <Suiryc> (even if the chunk is shorter) [21:59] <Suiryc> so for each chunk it find there is 1 frame in it [21:59] <Suiryc> which is really the case (each mp3 frame is in its [21:59] <Suiryc> so WMP got the correct number of mp3 frames played [22:00] <Suiryc> and since it has the correct rate (each frame conta [22:00] <Suiryc> it also got the correct timecode for the frame [22:00] <Belgabor|Home> ok [22:00] <Suiryc> resulting in a perfectly synched MP3 stream [22:01] <Suiryc> I was lead to this conclusion without debugging WMP [22:02] <Suiryc> I changed the dwScale value (with or without the nB [22:02] <Suiryc> but this resulted in otu of synch issues (audio pla [22:02] <Suiryc> out* [22:02] <Suiryc> I changed the nBlockAlign valuie : [22:03] <Suiryc> setting it to 1 and then I have out of synch issues [22:03] <Suiryc> but setting it 2304 and I stil have a perfectly syn [22:03] <Belgabor|Home> ok [22:04] <Suiryc> so in fact the 1152 value in nBlockAlign could be a [22:04] <Suiryc> _but_ [22:04] <Suiryc> must be higher than the size of an mp3 frame [22:04] <Belgabor|Home> ok, what happens if you set it to 0? [22:04] <Suiryc> lol [22:05] <Suiryc> if you set it to 0 then WMP won't play the stream ([22:05] <Suiryc> so no VBR ;) [22:05] <Belgabor|Home> ok [22:06] <Belgabor|Home> so the failure is in priciple not in avi, bu [22:06] <Suiryc> yep [22:06] <Suiryc> but since the AVI will use WAVEFORMATEX for audio h [22:07] <Belgabor|Home> do you have the resemblance of an idea why v [22:07] <Belgabor|Home> yep [22:07] <Suiryc> <Belgabor|Home> do you have the resemblance of an i [22:08] <Belgabor|Home> yep, why it fails sometimes [22:08] <ChristianHJW> thats what i am interested in also [22:08] <Suiryc> well in the case of WMP, it will divide the chunk s [22:08] <Suiryc> (that's what I think, since the synch is good) [22:08] <Suiryc> and will set it to 1 if the chunk size is too small [22:09] <Suiryc> but there is another way to compute timecode [22:09] <Suiryc> (assuming that you have CBR of course) [22:09] <Suiryc> you take the total bytes in previous chunks [22:09] <Suiryc> and divide it by nblockAlign [22:10] <Belgabor|Home> which fails miserably for the vbr hack [22:10] <Suiryc> of course in this case you get a completly wrong va [22:10] <Suiryc> yep [22:10] <Suiryc> otehr tools may also assume that the chunk is not v [22:11] <Belgabor|Home> ok, thats the failure in principle, but why [22:12] <Suiryc> what files ? [22:12] <Suiryc> broken ? what do you mean by broken ? [22:13] <Belgabor|Home> i had some vbr mp3 avis which seemed like ha

[22:13] <Suiryc> dunno [22:13] <Suiryc> maybe a problem with the decoder [22:14] <Belgabor|Home> ok, well that cleared things up a bit [22:14] <Belgabor | Home> thx :) [22:14] <Suiryc> :) [22:14] <Suiryc> btw there may be problems with Nandub code ;) [22:14] <Suiryc> because : [22:14] <Suiryc> 1. layer1 streams only have 384 samples per frame [22:15] <Suiryc> 2. IIRC with very high bitrates an mp3 frame can be [22:15] <Suiryc> s/higher/bigger [22:16] <Suiryc> (the max size is near 2000 bytes IIRC) [22:16] <Belgabor|Home> ok, so nBlockAlign should be >2000 [22:17] <Suiryc> so depending on the way dividing is used (rounding [22:17] <Suiryc> and the max size of a frame, it may find there are [22:17] <Belgabor|Home> ok, i got that [22:18] <Suiryc> but this is for really high bitrates ... [22:18] <Suiryc> lemme check ... [22:19] <Belgabor|Home> what would happen if we put two frames in on [22:20] <Belgabor|Home> no, not two, just double the values? [22:21] <Suiryc> if you double the value the rate of the audio will [22:21] <Suiryc> so to keep it correct you would have to put 2 mp3 f [22:22] <Suiryc> but then you would most likely go beyond the 1152 b [22:22] <Suiryc> and increase the chances to generate out of synch p [22:23] <Belgabor|Home> let me rethink [22:24] <Suiryc> changing dwRate and dwScale only affects the rate o [22:25] <Suiryc> multiplying dwRate by 2 => audio play 2 times faste [22:25] <Suiryc> multiplying dwScale by 2 => audio play 2 times slow [22:25] <Belgabor|Home> if we double dwrate, dwscale, nblockalign an [22:25] <Suiryc> multipyling both => no change [22:25] <Suiryc> dwSampleSize is set to 0 [22:26] <Belgabor|Home> ah ok, so skip that [22:26] <Suiryc> (dwRate, dwScale) and nBlockAlign are not linked [22:26] <Suiryc> you can use a higher value in nBlockAlign [22:27] <Suiryc> (like the 2304 I tested) [22:27] <Belgabor|Home> nvertheless, if we double all three, shouldn [22:27] <Suiryc> this won't change anything in the case of WMP becau [22:27] <Suiryc> Belgabor|Home : this would be safer [22:28] <Suiryc> but would cause even more troubles in apps that don [22:28] <Suiryc> I think some apps sometimes check a value of 1152 t [22:28] <Belgabor|Home> ok, i see the point [22:29] <Belgabor|Home> faulty concept stays faulty [22:33] <Suiryc> k I checked [22:33] <Suiryc> keeping 1152 shoudln't cause too much problems [22:33] <Suiryc> for Mpeg1-Layer2/3 the mas is near 1750 bytes long [22:34] <Suiryc> there could be problems with Mpeg2/2.5-layer2/3 [22:34] <Suiryc> where a 160kbps stream of 8kHz have frames of 2881 [22:35] <Suiryc> anyway I don't think people use this kind of stream [22:36] <ChristianHJW> highly unlikely ... [22:51] <Suiryc> nite [22:52] * Suiryc has left #matroska

The central problem is (imho) it works now. As soon as M\$ changes the rounding function in their avi parser from ceil to something else, it will stop working.