FFCoder

[**FF**mpeg En**coder** GUI]

An audio/video transcoding tool for windows

© 2011 Tony George

Help Document

Written by
Tony George

Homepage
http://www.teejee2008.wordpress.com/ffcoder

Email
tony.george.kol@gmail.com

Discussion
http://forum.doom9.org/showthread.php?t=149452

Table of Contents

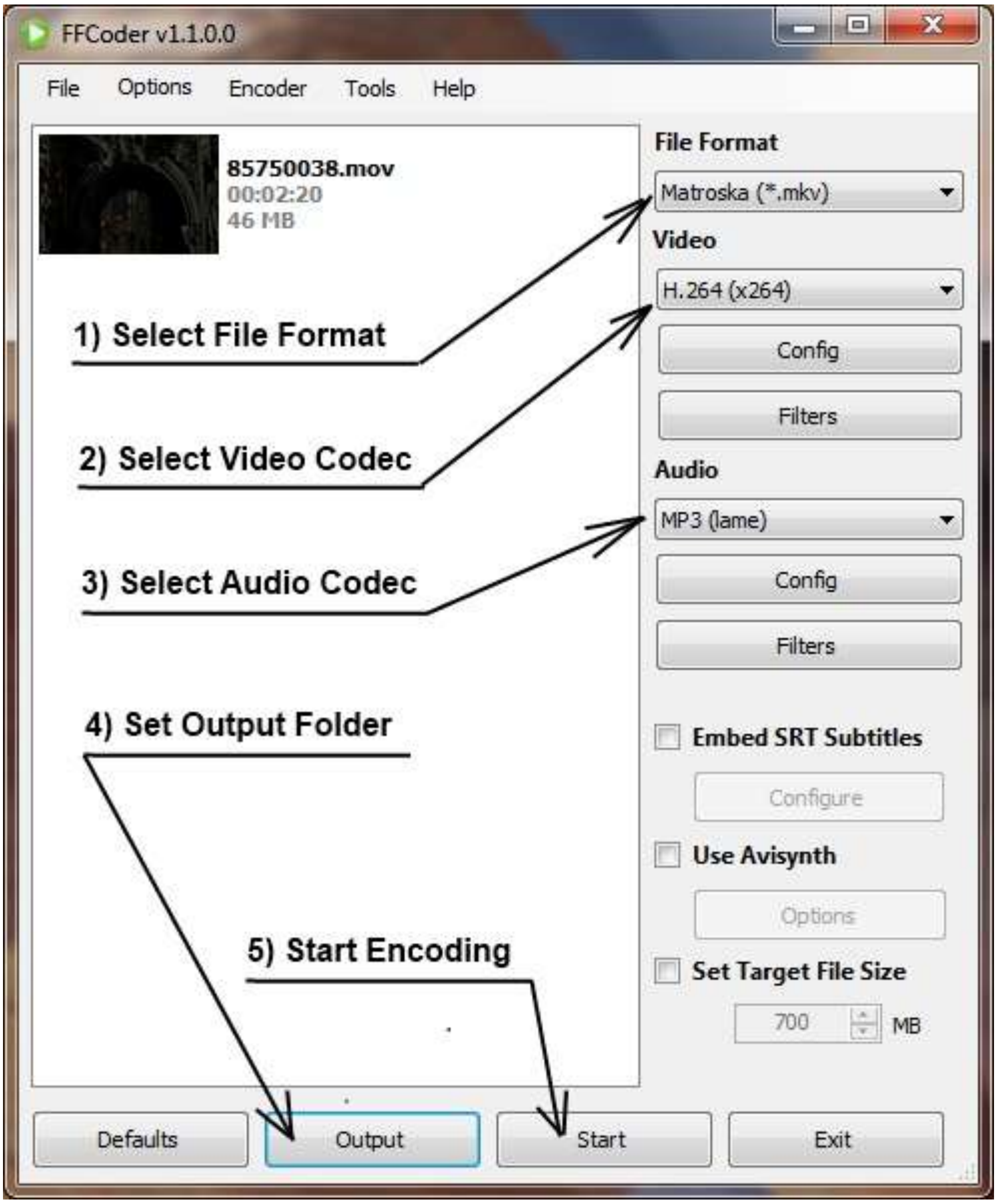
1. Introduction

FFCoder is a .NET based audio/video converter, which is capable of transcoding audio and video files to various formats. It is a unified frontend for many freeware utilities such as FFmpeg, MPlayer and MEncoder.

- Supports 16 command line encoders : FFmpeg, MEncoder, FFmpeg2theora, x264, XVID EncRaw, avs2avi, LAME, FAAC, Nero AAC, OggEnc, WavPack, MusePack, Monkey's Audio, FLAC, Speex and OptimFROG
- Each encoder can be used alone or in combination with other encoders.
- Batch processing
- Parallel processing (multiple files can be transcoded at the same time)
- Fully unattended directory watch mode
- Advanced configuration options for all codecs.
- Supports 2-pass encoding with automated first pass.
- MeGUI presets can be imported for x264 and xvid encoding.
- Input files can be avisynth scripts or any format supported by ffmpeg.
- Tools: Dump audio/video stream to file, create images in JPEG/BMP/PNG/TIFF format, rebuild/cut/join video files, quantization matrix editor, etc.

Even though this program is not targeted at beginners, every effort has been made to keep the GUI as simple as possible. A basic understanding of audio/video formats is all that is required.

2. The Main Window



85750038.mov
00:02:20
46 MB

File Format: Matroska (*.mkv)
Video: H.264 (x264)
Audio: MP3 (lame)
☐ Embed SRT Subtitles
☐ Use Avisynth
☐ Set Target File Size 700 MB

1) Select File Format
2) Select Video Codec
3) Select Audio Codec
4) Set Output Folder
5) Start Encoding

Basic steps:

0) **Add the files to convert –** Click on File->Add Files from the main menu or drag files on the main window to add them to the input list.

1) **Select the file format** – Select the required output file format like AVI, MKV, MP4, etc. Default audio and video codecs will be automatically selected.

2) **Select the video codec** – Select the video codec from the drop-down list. Click on ‘Config’ for setting the bitrate and configuring the codec. Click on ‘Filters’ to change video properties like frame rate, frame size, etc.

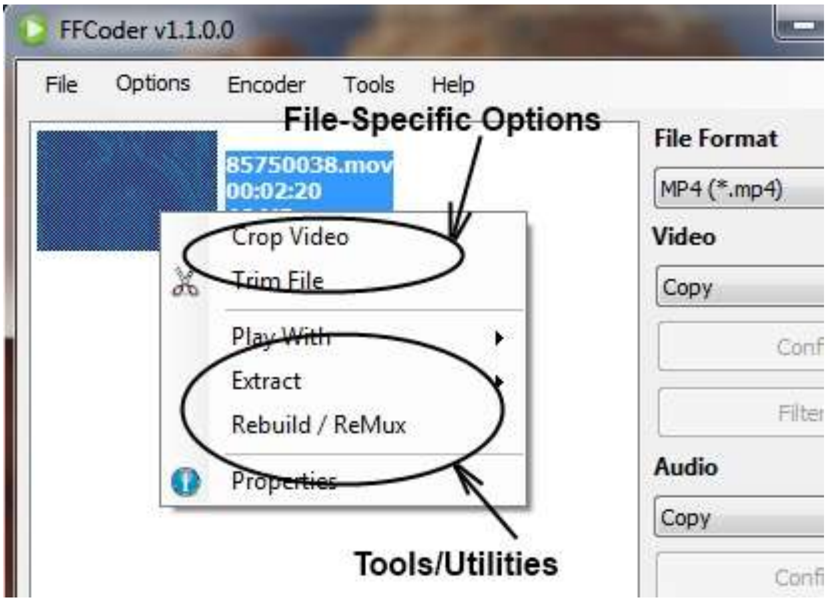
3) **Select the audio codec** – Select the audio codec from the drop-down list. Click on ‘Config’ for setting the bitrate and configuring the codec. Click on ‘Filters’ to change audio properties like sampling rate and number of channels, etc.

4) **Select the output folder** – Select the folder where you want to save the converted files.

5) **Click on Start** to begin encoding your files.

2.1 The Right-click menu

A lot of options are available by right-clicking a file in the input list.



The **file specific options** apply to a **single file** only. All other options apply to the **whole batch**. If there is a conflict between an option that applies to a file and also applies to the whole batch, then the file-specific setting has greater priority and will be used for encoding *that* file.

The tools are simple utilities that can be used on a file. They do not affect the main encoding process. Some of the tools available are given below:

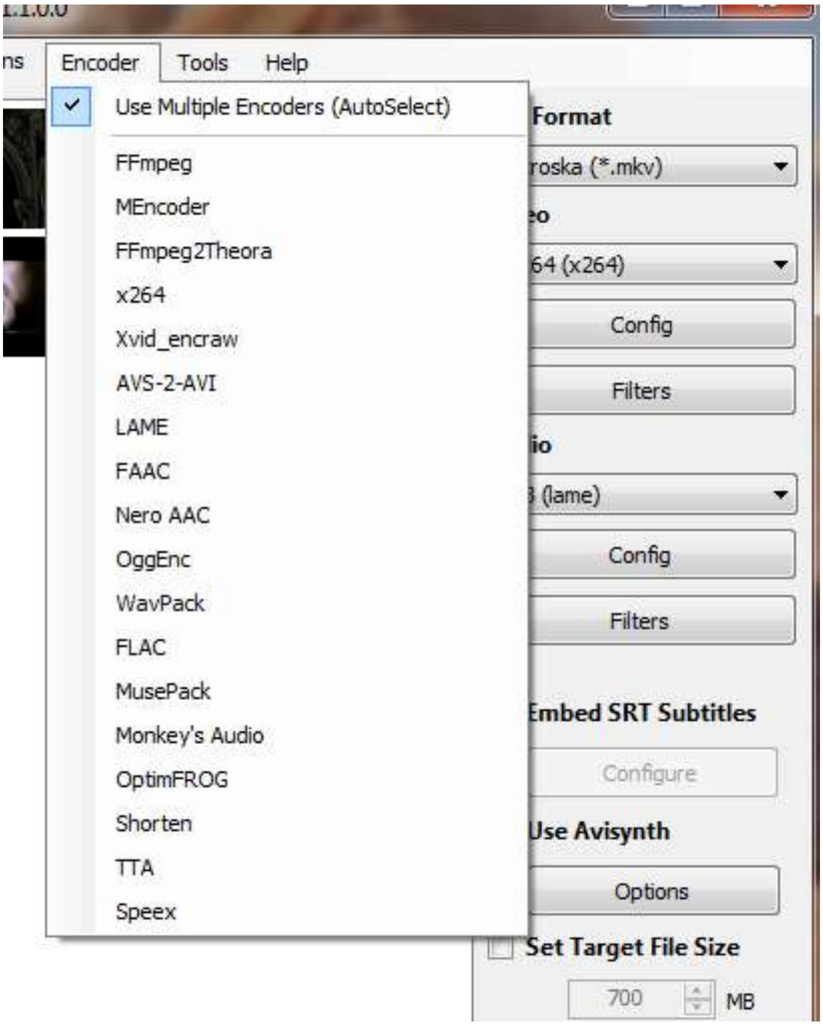
- **Extract:** The extraction tool can be used for extracting the audio/video tracks from a file without re-encoding. Simply select an audio/video track from the drop-down menu and specify the file to save.

- **Play:** Play the input file using various media players like MPlayer, FFPlay, Media Player Classic, or your default media player.

- **Rebuild/Remux:** The rebuild tool simple copies the audio and video tracks from the input file and saves is in the output file. This is useful for:
    - Quickly changing the file format without re-encoding or quality loss.
    - Fixing partially downloaded/broken video files.
    - Creating MKV files with embedded subtitles. Simply place the SRT file in the same folder as the input file and select MKV as the output format.

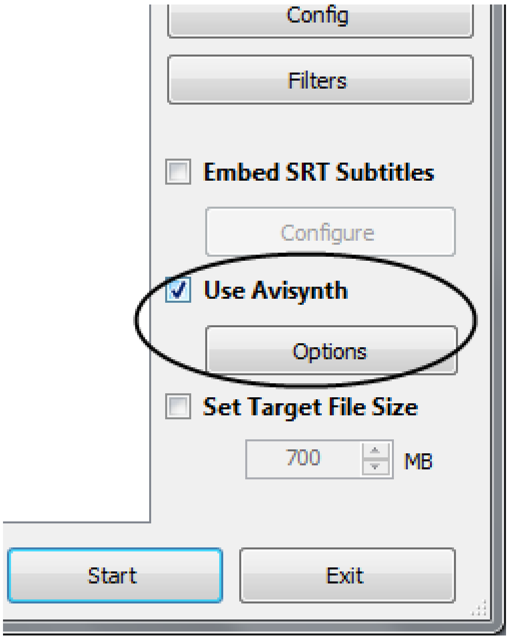
- **Properties:** Displays the audio/video properties for the selected file.

2.2 The Encoder menu



The Encoder menu option lets you select an encoder. An encoder is a command line tool that does the actual encoding. FFCoder is simply a GUI (Graphical interface) for these encoders. Each encoder can encode files in various formats using various codecs and filters. The options available depend on the selected encoder.

2.3 Avisynth Mode



☐ Embed SRT Subtitles

☑ Use Avisynth

☐ Set Target File Size

700 MB

The basic idea of Avisynth is this:

- Download and install Avisynth on your system
- Create a simple text file with an '.avs' file extension.
- Write a script for opening a media file and/or modifying the audio and video
- The AVS file can now be played as a normal media file in Windows Media Player or any other player. The player sees the AVS file as any other media file and is not aware that the audio/video data is coming from avisynth.

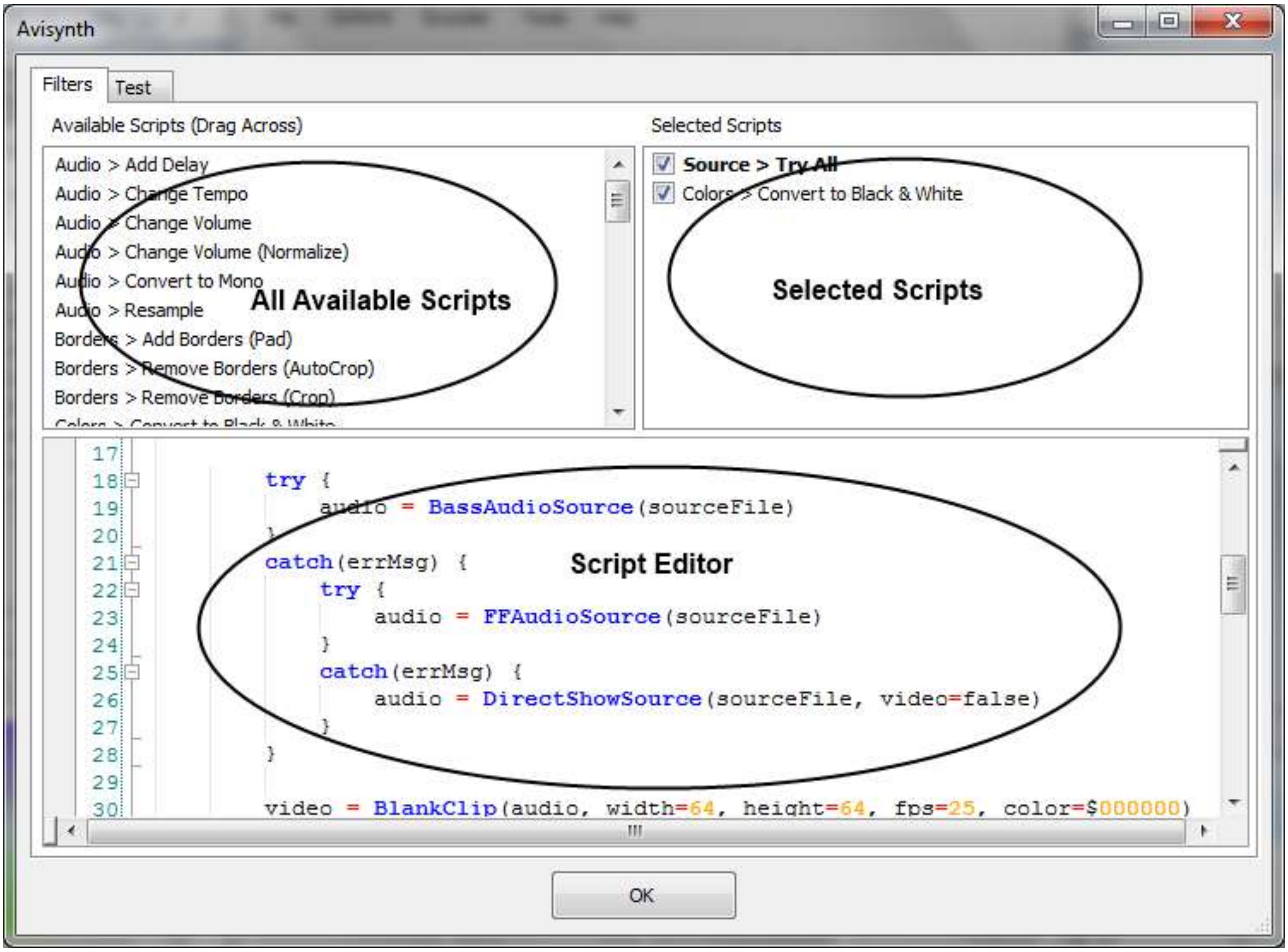
Avisynth is extremely useful for modifying the audio and video in the input file. You can clean up damaged videos by removing noise and jitter, apply various effects, etc.

Avisynth processing can be enabled by checking the **Use Avisynth** option in the main window. When this option is selected FFCoder will create an avisynth script file (*.avs) for each input file. This AVS file is then encoded to give the final output. Avisynth provides a large number of filters for processing audio and video. Click **Options** to customize the avisynth script.

2.4 Target File Size

Depending on the selected encoder and the selected video codec, you have the option to set the target file size for the converted files. The video will be encoded in 2 passes and the bitrate will be calculated automatically.

3. Avisynth Options


Avisynth
Filters
Test
Available Scripts (Drag Across)
Audio > Add Delay
Audio > Change Tempo
Audio > Change Volume
Audio > Change Volume (Normalize)
Audio > Convert to Mono
Audio > Resample
Borders > Add Borders (Pad)
Borders > Remove Borders (AutoCrop)
Borders > Remove Borders (Crop)
Colors > Convert to Black & White
All Available Scripts
Selected Scripts
☑ Source > Try All
☑ Colors > Convert to Black & White
Selected Scripts
Script Editor
OK

```
        try {
            audio = BassAudioSource(sourceFile)
        }
        catch(errMsg) {
            try {
                audio = FFAudioSource(sourceFile)
            }
            catch(errMsg) {
                audio = DirectShowSource(sourceFile, video=false)
            }
        }

        video = BlankClip(audio, width=64, height=64, fps=25, color=$000000)
```

There are two lists displayed in the options window:

1) A **master list** on the left-hand-side containing all available filters.
2) A **list of selected filters** on the right-hand-side which displays all currently selected filters. This list will be saved in the preset file (if you decide to create a preset).

Managing the filter lists

- ✓ A filter can be moved from one list to the other by selecting it and dragging it with the mouse. Dragging an item from the Master list to the Selected-filters-list will create a copy. Dragging an item from the Selected-filters-list to the Master list will move the item. Hold down CTRL to create a copy.
- ✓ In each list the items can be moved up and down by dragging with the mouse. The order of items is important since the filters are applied in the same order.
- ✓ To create a copy of the selected filter(s), hold down CTRL while dragging the item up or down.
- ✓ To delete the selected filter(s) press the DELETE key on the keyboard or do Right-click → Remove

Editing the filter Scripts

Each item in the list is a simple avisynth script. These scripts are combined together to create the final script file that will be used for encoding. You can edit each item by simply selecting the item and editing the contents in the script editor.

Things to keep in mind:

- ✓ Each item in the list is a simple avisynth script. You can do everything that you normally do within an avisynth script.
- ✓ The **Gscript** plugin is loaded by default. The Gscript plugin extends the avisynth scripting language by providing if-then-else, looping constructs, etc (More info here).
- ✓ Some variables are created by default that you can use in your scripts:

VariableName	**Type**	**Value**
sourceFile	String	Path to input file
sourceExt	String	File extension of source file (in lowercase and without the dot)
sourceFPS	Number	Frames/second of input video (fractional value, as detected by MediaInfo)
sourceIsInterlaced	Boolean	True if the source video is detected as interlaced by MediaInfo.
sourceHasOnlyAudio	Boolean	True if the input file contains only audio and no video track.
sourceHasOnlyVideo	Boolean	True if the input file contains only video and no audio track.
sourceAudioTracks	Number	The number of audio tracks in the input file
sourceVideoTracks	Number	The number of video tracks in the input file
plugins	String	The path to FFCoder's Plugins directory

- ✓ Some functions are created by default that you can use in your scripts:

Function	**Description**
AutoYV12	Checks if the video clip is in YV12 color space and converts it.
AutoRGB32	Checks if the video clip is in RGB32 color space and converts it.

- ✓ Plugins that are used in the scripts can be put in Avisynth's **Plugins** folder from where it will be autoloaded by Avisynth, or you can put the plugins in the Plugins directory in the application folder and load them manually using the following syntax:

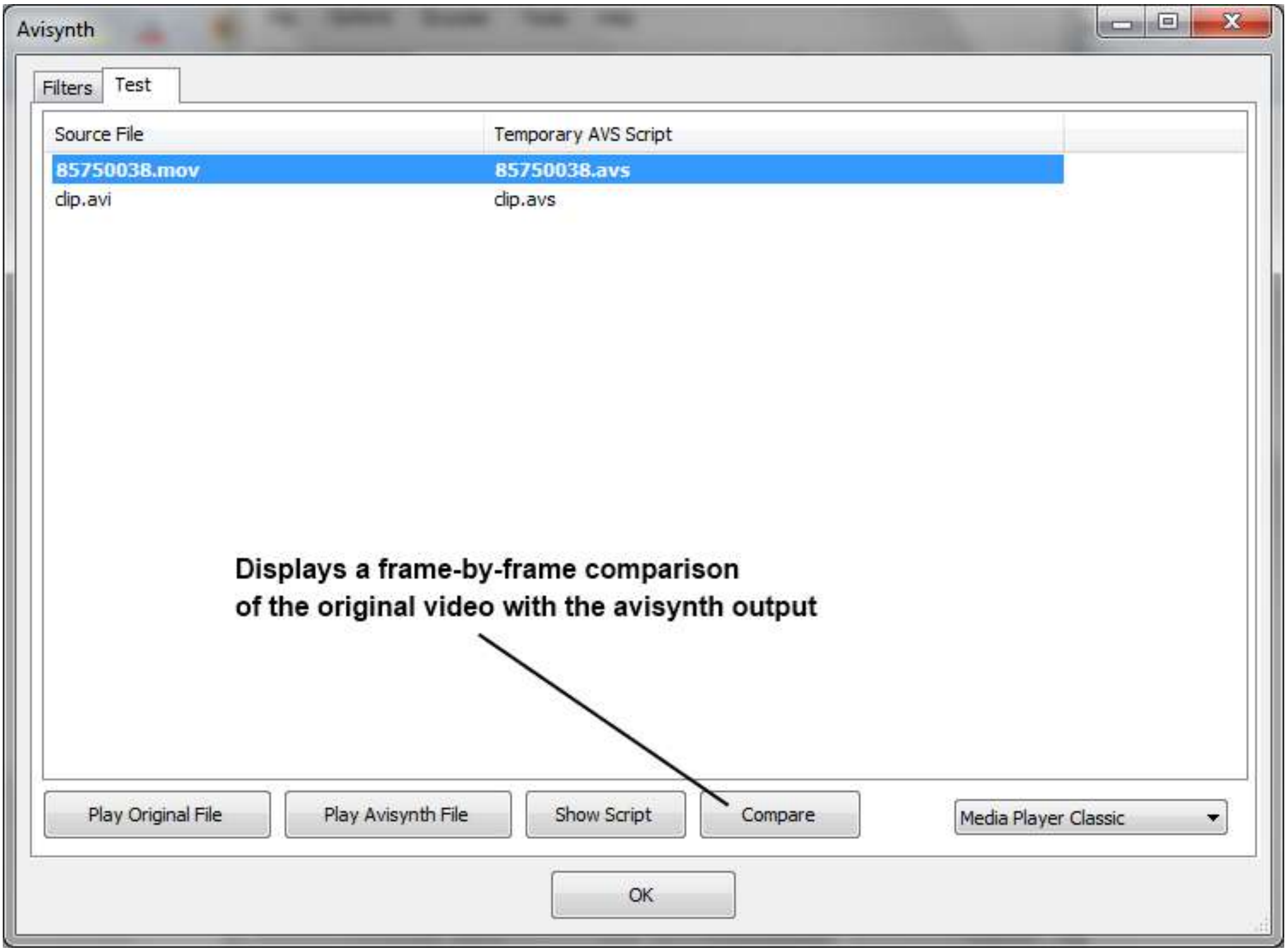
```
LoadPlugin(plugins + "hqdn3d.dll")

c = c.AutoYV12()
c = c.hqdn3d()
```

- ✓ From v1.2 onwards, it is no longer necessary to add a source filter for opening the input file. A source filter will be automatically inserted. You must open the input file using a source filter before trying to apply other filters.

- ✓ Be careful while editing the script for an item. Errors in the script will cause encoding to fail and can be difficult to detect and to remove. For complex tasks please create an Avisynth script and use it as the input file (like you normally do).

3.1 Testing the scripts



Source File	Temporary AVS Script
85750038.mov	85750038.avs
clip.avi	clip.avs

Displays a frame-by-frame comparison of the original video with the avisynth output

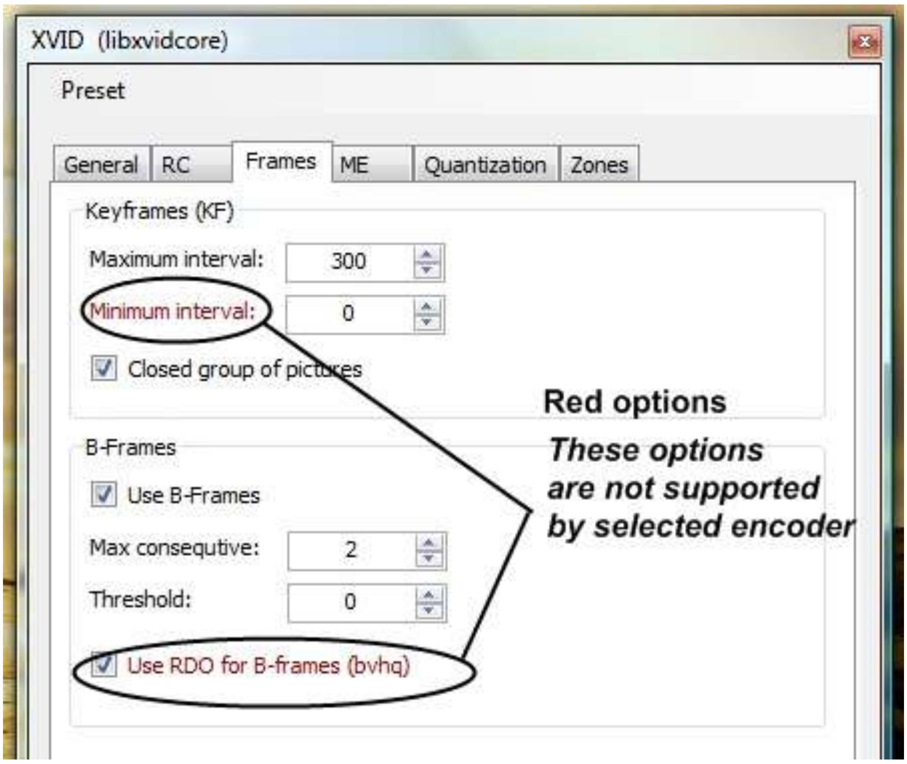
- ✓ The **Test** tab provides some options for testing the scripts. It displays the list of temporary scripts that will be generated.
- ✓ **Play Original File:** Plays the original (source) file with Media Player Classic.
- ✓ **Play Avisynth File:** Plays the AVS file with Media Player Classic.
- ✓ **Show Script:** Displays the script that will be created for encoding the file.
- ✓ **Compare:** Displays a frame-by-frame comparison of the original file with the AVS file

4. Codec configuration windows

4.1 Tool tips

Just move your mouse over an **option label , radio button** or **checkbox** to see a description. Most of these desciptions are from the MPlayer documentation and some are from various places on the Internet.

4.2 Unsupported Options



Maximum interval: 300

Minimum interval: 0

☑ Closed group of pictures

☑ Use B-Frames

Max consequtive: 2

Threshold: 0

☑ Use RDO for B-frames (bvhq)

Red options

These options are not supported by selected encoder

XVID configuration window when FFmpeg is selected as encoder

Sometimes you will notice that some options are highlighted in red on the codec configuration window. These options are not supported by the currently selected encoder. In other words the selected encoder doesn't have a command line switch for changing the option.

For example, if you select **FFmpeg** from the **Encoder** menu, select the **XVID** video codec and open the configuration window you will notice that some options are highlighted in red (image above). FFmpeg does not have a command line option (or switch) for setting the value of **Minimum Keyframe Interval** and **BVHQ** options. Therefore, these options will remain at their default value even if you change the values from the GUI.

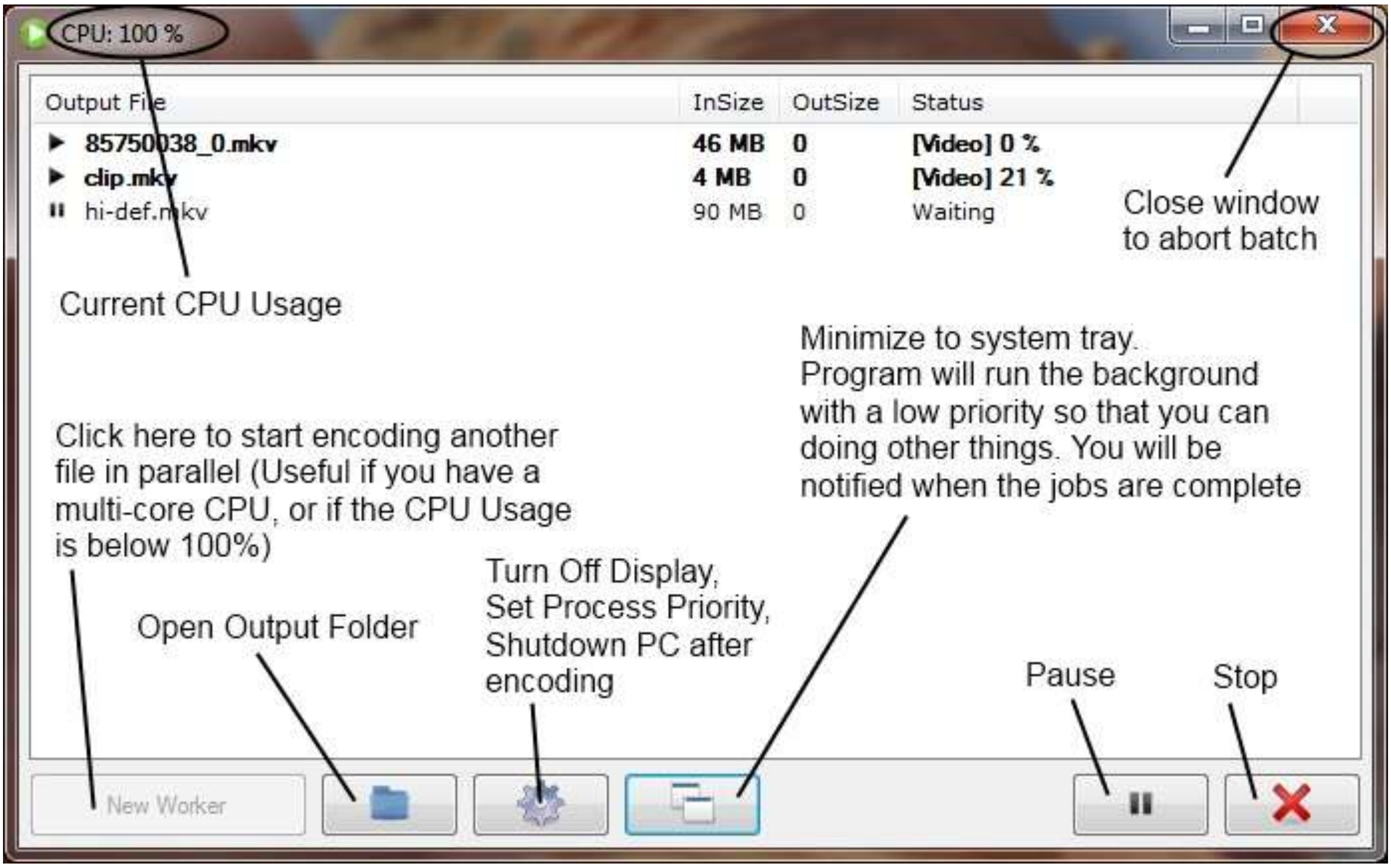
If you want to use these options then use another encoder which supports them. For example **MEncoder** supports both **Minimum Keyframe Interval** and **BVHQ** options.

Please Note:

- Just because FFmpeg doesn't provide a command line switch for some options, it does not mean that FFmpeg is inferior to other encoders or that the video produced is of poor quality. It only means that you cannot change the **default value**. Most of these options are **advanced options** which can degrade the video quality if set to a bad value. There is usually no use in tweaking these options, unless you are an expert and know what you are doing.

- The codec configuration options provided by this program are very exhaustive. Some of the video codecs have dozens of parameters and these settings can be a little intimidating for first-time users. Please remember that just because an option has been made available to you, it doesn’t mean that you have to change it in order to get better quality. All the default settings are already optimized for the best speed/quality trade-off. You only need to select an output format and select the output folder.

5. The Progress window



Output File	InSize	OutSize	Status
▶ **85750038_0.mkv**	**46 MB**	**0**	**[Video] 0 %**
▶ **clip.mkv**	**4 MB**	**0**	**[Video] 21 %**
⏸ hi-def.mkv	90 MB	0	Waiting

Close window to abort batch

Minimize to system tray. Program will run the background with a low priority so that you can doing other things. You will be notified when the jobs are complete

Click here to start encoding another file in parallel (Useful if you have a multi-core CPU, or if the CPU Usage is below 100%)

Turn Off Display, Set Process Priority, Shutdown PC after encoding

5.1 Parallel processing

If you have a multi-core CPU you can encode many files at the same time to fully utilize your CPU and to speed up encoding. Click on the **New Worker** button to start a new worker thread that will start working on another file.

Running many jobs at same time is useful only if the audio/video codec is not multi-threaded. For example, when you encode files using the **x264** video codec, there is no need to create a new worker since x264 is multi-threaded and is able to fully utilize your CPU (CPU usage will be 100%).

This button will be disabled in the following cases:

- If you have already created maximum number of workers. For example, if you have a dual-core CPU you can only create two workers.
- Some files may be shared/locked by the encoder process because of which it may not be possible to encode multiple files at same time.
- If the video encoder is multi-threaded then then is no use of running multiple jobs

5.2 Pause, Resume and Abort

You can **Pause** or **Abort** the batch by clicking on the **Pause** or **Stop** button. It is not possible to pause or abort a single file. You have to pause or abort the whole batch. Trying to pause or abort a single file is problematic because of which it has not yet been implemented. This option will be added in the future.

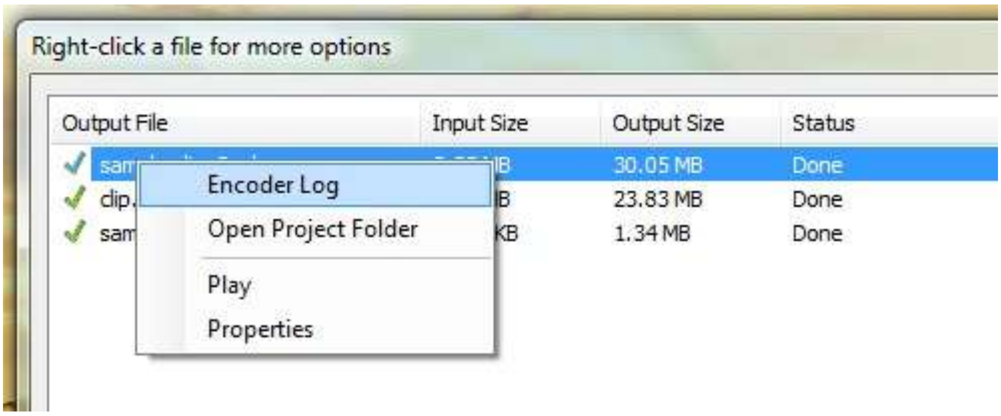
5.3 Progress Info

Progress information is displayed in columns. The information is obtained by parsing the console output produced by the encoder.

- Output filename
- Input file size
- Output file size (Current or Estimated) (If reported by encoder)
- Encoding status
- Encoding speed in frames/second (If reported by encoder)
- Audio/video bitrate (If reported by encoder)
- Estimated time remaining (If reported by encoder)

The info that is displayed depends on the encoder. For example, if an encoder reports the **frames/sec** value during encoding, it will be displayed in the **FPS** column - otherwise the column will be empty.

5.4 The Right-click menu



You can right-click on any file after the files have been converted to get a right-click menu.

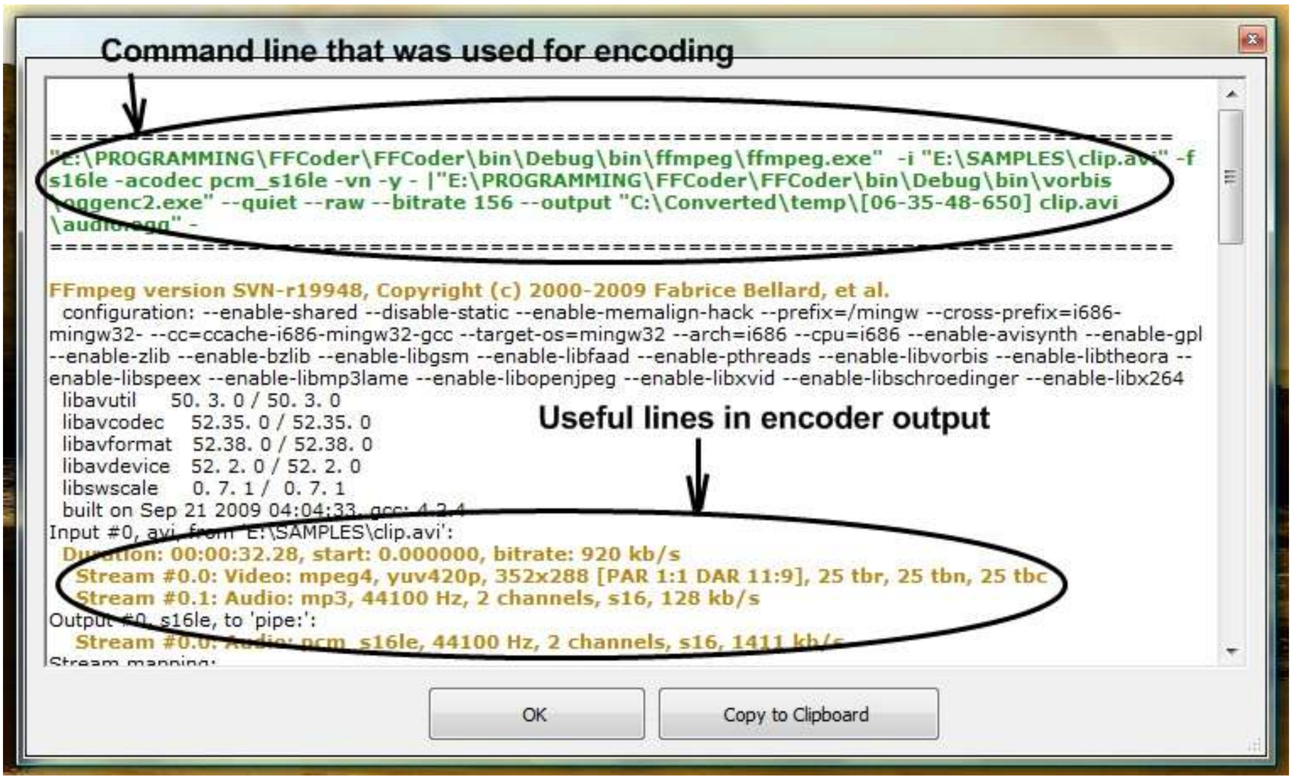
- **Encoder log** – Click here to view a summary of the command line that was used for encoding and the console output produced by encoder. Useful for viewing the error message reported by the encoder after encoding fails.

- **Open Temp folder** – Opens the temporary folder which contains the batch scripts that were used for encoding and other files that may be useful to experienced users.

- **Play** – Play the output file using the in-built player. Useful for checking the video quality after encoding.

- **Properties** – View the file information provided by MediaInfo. Useful for checking if the file produced is okay.

6. The Encoder Log


Command line that was used for encoding
Useful lines in encoder output
OK
Copy to Clipboard

```
====================================================================================
"E:\PROGRAMMING\FFCoder\FFCoder\bin\Debug\bin\ffmpeg\ffmpeg.exe"  -i "E:\SAMPLES\clip.avi" -f
s16le -acodec pcm_s16le -vn -y - |"E:\PROGRAMMING\FFCoder\FFCoder\bin\Debug\bin\vorbis
\oggenc2.exe" --quiet --raw --bitrate 156 --output "C:\Converted\temp\[06-35-48-650] clip.avi
\audio.ogg" -
====================================================================================

FFmpeg version SVN-r19948, Copyright (c) 2000-2009 Fabrice Bellard, et al.
  configuration: --enable-shared --disable-static --enable-memalign-hack --prefix=/mingw --cross-prefix=i686-
mingw32- --cc=ccache-i686-mingw32-gcc --target-os=mingw32 --arch=i686 --cpu=i686 --enable-avisynth --enable-gpl
--enable-zlib --enable-bzlib --enable-libgsm --enable-libfaad --enable-pthreads --enable-libvorbis --enable-libtheora --
enable-libspeex --enable-libmp3lame --enable-libopenjpeg --enable-libxvid --enable-libschroedinger --enable-libx264
  libavutil     50. 3. 0 / 50. 3. 0
  libavcodec    52.35. 0 / 52.35. 0
  libavformat   52.38. 0 / 52.38. 0
  libavdevice   52. 2. 0 / 52. 2. 0
  libswscale     0. 7. 1 /  0. 7. 1
  built on Sep 21 2009 04:04:33, gcc: 4.2.4
Input #0, avi, from 'E:\SAMPLES\clip.avi':
  Duration: 00:00:32.28, start: 0.000000, bitrate: 920 kb/s
    Stream #0.0: Video: mpeg4, yuv420p, 352x288 [PAR 1:1 DAR 11:9], 25 tbr, 25 tbn, 25 tbc
    Stream #0.1: Audio: mp3, 44100 Hz, 2 channels, s16, 128 kb/s
Output #0, s16le, to 'pipe:':
    Stream #0.0: Audio: pcm_s16le, 44100 Hz, 2 channels, s16, 1411 kb/s
Stream mapping:
```

After a file has finished encoding, you can right-click on the file and view the encoder log. The encoder log displays the command string that was used for encoding the file and the messages generated by the encoder.

Colors

The **command line** is highlighted in **Green**
Error messages are highlighted in **Red**.
Other **useful lines** are highlighted in **Yellow.**

7. Presets

There is no need to choose the settings manually each time you use the program. You can save the settings to a preset file and load them again whenever you need it. Presets are saved as XML files in the **\presets** folder.

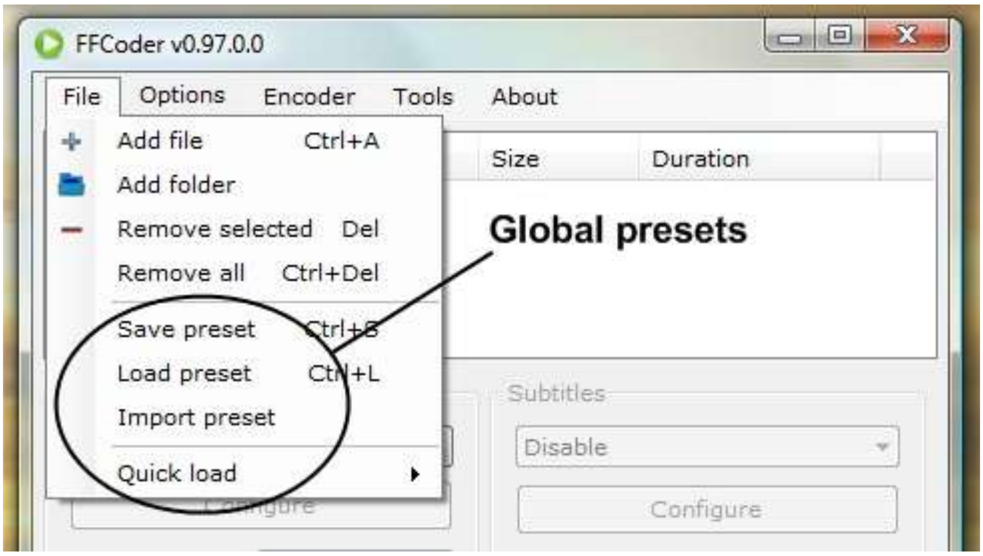
There are two types of presets used by this program:

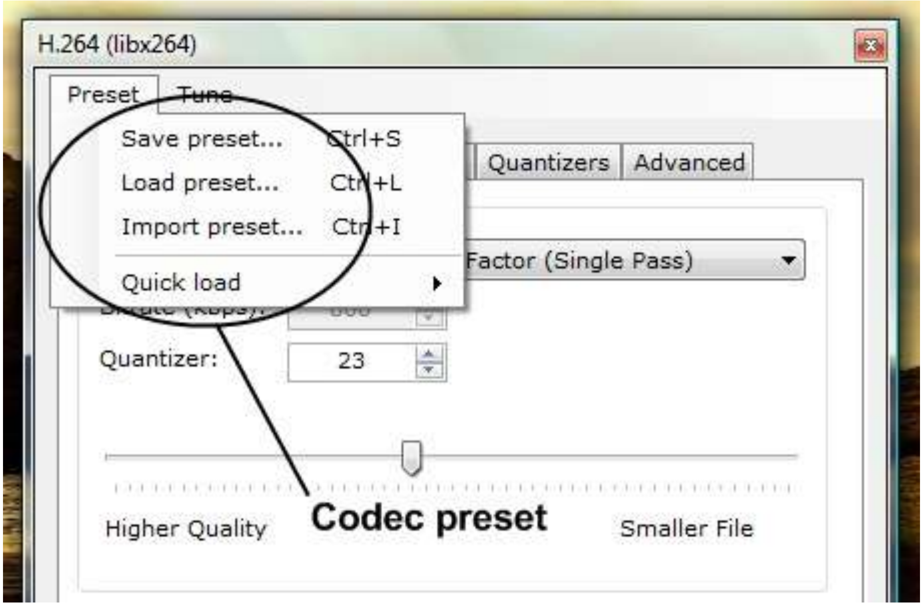
1) **Codec-level presets:** These presets can be created/loaded from the **Presets** menu in the codec configuration window. They store settings for a particular codec only and are currently available for XVID, x264 and Lavc video codecs (The remaining codecs have very few settings). These presets are saved in the **\presets\codec\<codec-name>** folder.

2) **Global presets:** These presets can be created/loaded from the **File** menu in the main window. Global presets store the following settings:

   - ✓ Encoder name
   - ✓ File format name, file extension and muxer settings
   - ✓ Video codec name and codec settings
   - ✓ Audio codec name and codec settings
   - ✓ Video and audio filter settings

Note: **File-specific options** such as file names, duration, cropping values, track selection are not stored in global presets. So each time you add a file to the list you have to set these options manually.





7.1 Importing presets

MeGUI presets can be imported for XVID and X264 encoding, using the **Import preset** option in the **Preset** menu. The preset file will be converted into FFCoder's preset format and saved in the **\presets** folder as an XML file. The preset will also be available in the **Quick load** menu.

You can also import presets from X-Media Recode using **Import preset** option in the **File** menu on the main window. X-Media Recode saves presets as an INI file. After importing, it will be converted to FFCoder's XML format and saved in the **\presets\global** folder.

7.2 Quick load menu

The **Quick load** menu is a quick way of loading a preset. It displays all presets found in the associated preset folder. For example, on the main window, the **Quick load** menu displays all presets found in the **\presets\global** folder. All sub folders are displayed as groups, so you can group your presets by putting them in sub folders.

7.3 Sessions

When the program exits, all settings are saved to a file named **LastSession.xml**. This file is loaded when the program is started the next time. This means that you can exit and start FFCoder anytime you want without losing settings. The only exception is that file settings are not kept. You will have to add the input files again.

8. The Temp folder

The temp folders are created in **<%Temp%>/ffcoder_temp** before encoding starts. A temporary folder is created for each input file. This folder is normally deleted when you exit the program but you can choose to keep this folder by changes the option in the Preferences window.

The project folder for each input file contains the following files:

- **Script-*.cmd** – These are batch files used for encoding.
- **Script-*-log.txt** – Text files which log the console output produced by each batch file.
- **Source.txt** – Info about source file.
- **Preset.xml** – The GUI settings that were used.
- ***.avs** – Temporary AVS file that was generated for encoding
- ***.avsi** – Temporary AVS file that was generated for encoding
- **Temp.wav** – Temporary WAV file produced from source file audio.
- **Audio.*** - Temporary audio file produced by encoder.
- **Video.*** - Temporary video file produced by encoder.

Notes:

1) The batch files are identical to the batch files used for encoding the files. If the encoding fails then you can open the project folder and examine the batch files that were used for encoding. You can execute them by double-clicking to see the error message.

2) The project folder contains temporary audio and video files that were created during encoding (Temporary files are created only if Encoder option is set to 'Use multiple (Auto-Select)'. These temporary files may be useful for experienced users. For example, you may want to grab the audio and video files from this folder and mux them yourself or process them with another program.

3) If encoding fails, the first place to check for errors is the command line string that was generated (Open the CMD files in Notepad to see the command line string). You can edit the script files to try and fix the problem and run the scripts to check if they work.

9. Supported Encoders

9.1 FFmpeg Encoder

A versatile audio/video transcoding tool. Supports the largest number of audio/video formats for input and output.

http://ffmpeg.mplayerhq.hu/

Pros:
- Can transcode audio/video files to a large number of audio/video formats.
- Large number of audio/video codecs.
- Supports Avisynth files as input.

Cons:
- Basic filters only like crop, resize, etc.
- Almost no options for encoding subtitles.
- Copying the audio or video stream without re-encoding is problematic.

9.2 MEncoder (Movie Encoder)

http://www.mplayerhq.hu/

Pros:
- Large number of useful audio and video filters.
- The effect of applied filters can be previewed with MPlayer.
- Subtitles can be **rendered** onto the video (hard subtitles).
- Supports Avisynth files as input.
- Best command line encoder for XVID encoding. Supports almost all XVID options including device profiles.

Cons:
- Cannot transcode audio files. The input and output files **must contain video**.
- Can create **AVI** and **MPEG** files only. It can also encode to other formats like MKV, FLV etc but the files produced are problematic. The files produced may be difficult to seek.
- No option for **embedded** subtitles (soft subs).

9.3 FFmpeg2Theora

The best command line tool for creating OGG Theora videos.

http://v2v.cc/~j/ffmpeg2theora/

Pros:
- Regularly updated to include the latest Theora library.
- Has a good number of video filters like deblock, deinterlace, etc.
- Supports embedded subtitles. SRT subtitles can be encoded as a separate stream.

Cons:
- Does not support Avisynth files as input.

9.4 X264 Encoder

The best command line tool for creating H.264 videos.

http://www.videolan.org/developers/x264.html
http://mirror05.x264.nl/Sharktooth/

Pros:

- It is the official x264 encoder and hence has the latest features and improvements.
- Supports a large number of input formats via libavcodec
- Supports Avisynth files as input.
- Supports AVI, MKV and MP4 output.

Cons:

- Does not have a filter for setting the output framerate

9.5 Xvid EncRaw Encoder

A command line encoder for XVID encoding. It is recommended to use MEncoder for XVID encoding.

http://www.xvid.org/

Pros:

- Nothing in particular.

Cons:

- Does not support all XVID options.
- Supports only Avisynth files as input (It supports piping but it's problematic)
- A little buggy
- Issues with AVI files over 2 GB?

9.6 AVS 2 AVI

A command line tool for encoding videos using VFW (Video for windows) codecs installed on your PC (Like DivX 7, VP6/VP7, FFDShow, etc).

http://moitah.net/

Pros:

- Encode video using VFW (Video for windows) codecs installed on your PC.
- Useful for encoding video using external codecs like DivX 7, VP6/VP7, FFDShow, etc.

Cons:

- Supports only Avisynth files as input.
- Creates AVI files only.

9.7 LAME MP3 Encoder

A command line tool for creating MP3 files.

http://lame.sourceforge.net/

Pros:
- Good quality.
- Supports CBR, ABR and VBR modes.

Cons:
- None.

9.8 FAAC AAC Encoder

A command line tool for creating MP4 audio files using the FAAC AAC codec. It is recommended to use Nero AAC encoder for AAC encoding, which gives much better quality, and supports more AAC features.

http://www.audiocoding.com/

Pros:
- Nothing in particular

Cons:
- Does not support many AAC profiles/features.
- Does not support High Efficiency (HEv1, HEv2) profiles

9.9 Nero AAC Encoder

A free command line tool from Nero for creating MP4 audio files. Currently the best AAC encoder available.

http://www.nero.com/enu/technologies-aac-codec.html

Pros:
- Good quality.
- Supports all AAC profiles and features.

Cons:
- None.

9.10 OggEnc (OGG Encoder)

A command line tool for creating OGG Vorbis audio files. It is recommended to use this for creating OGG Vorbis audio files. You can also use FFmpeg or FFmpeg2Theora for creating OGG Vorbis files.

http://www.vorbis.com/

Pros:
- Very good quality.

Cons:
- Encoding is around 2-3 times slower than the Vorbis codec implemented by FFmpeg

9.11 WavPack

A command line tool for creating WavPack (*.wv) audio files. It is a lossless format. So you can convert WAV files to WV (and back to WAV) without any quality loss.

http://www.wavpack.com/

Pros:

- Both lossless and lossy modes of operation.
- Good compression.

Cons:

- None.

9.12 FLAC (Free Lossless Audio Codec)

A command line tool for creating FLAC (*.flac) audio files. It is a lossless format. So you can convert WAV files to FLAC (and back to WAV) without any quality loss.

http://flac.sourceforge.net/

9.13 Musepack

The official command line tool for creating Musepack audio files (*.mpc). It is a lossy audio compression format based on the MPEG-1 Layer-2 / MP2 algorithms.

http://www.musepack.net/

9.14 Monkey's Audio

The official command line tool for creating Monkey's Audio files (*.ape). It is a lossless format. So you can convert WAV files to APE (and back to WAV) without any quality loss.

http://www.monkeysaudio.com/

9.15 OptimFROG

The official command line tool for creating OptimFROG audio files (*.ofr). It is a lossless format. So you can convert WAV files to OFR (and back to WAV) without any quality loss.

http://www.losslessaudio.org/

9.16 SpeexEnc

Speex is a **voice compression codec** from Xiph.org (makers of vorbis and theora codecs). It is open-source and patent free unlike other voice compression codecs like AMR (Adaptive Multi-Rate). SpeexEnc is the official encoding tool for speex.

http://speex.org/

10. Audio and video formats

- Formats like MusePack and Monkey's Audio are rare and not very widely used. You can download plugins for Winamp and Foobar for playing these formats on your PC. Visit their official websites for more information.

- Formats like FLAC, WavPack, OptimFROG, Monkey's Audio, True Audio, etc are lossless formats. They are useful for ripping audio CDs without any quality loss. The compression ratios are usually around 30-70% compared to lossy formats like MP3 which provide much better compression ratios at the cost of a little loss in quality. There is no question of choosing the audio bit rate while encoding with such codecs, since they are lossless. The options provided by such codecs are usually for selecting a trade-off between encoding speed and compression ratio. Some of these lossless codecs can work in lossy modes also.

- Audio codecs like AMR and Speex are **voice compression** codecs. These codecs are used for encoding voice recordings at very low bitrates with minimal loss in voice clarity. AMR is more widely supported (in mobile phones for example) but is not patent free. Speex is open-source and patent-free but not as widely used.

- Video codecs like FFv1, FFvHuff, HuffYUV, etc are **lossless codecs** which produce very large files. These are usually used for high quality video editing work or as a temporary format for storing video which will be encoded again.

11. Advantages

Everything that this program does can also be done by executing the command line tools from a command window. So why use this program? The advantages are given below:

1) Most encoders (LAME, OggEnc, etc) **support very few input formats**. This limitation is overcome by decoding the source file with FFmpeg and piping the decoded data to the encoder. So the input file can be in *any* format supported by FFmpeg for *any* encoder. For example, you can encode even AVI files (containing audio) with LAME MP3 Encoder.

2) Some encoders (like *ffmpeg2theora, LAME*) **do not support AVS files** as input. This limitation is overcome by decoding the AVS file with FFmpeg and piping the decoded data to the encoder. So the input file can be an AVS file for *any* encoder. For example, you can encode AVS files with ffmpeg2theora, LAME, etc.

3) Some encoders (like *avs2avi*) **support only AVS files** as input. It can be time consuming to manually create an AVS file for each file that you wish to encode. FFCoder automatically creates a temporary AVS file and feeds it to the encoder. You simply need to add the files to encode to the list and FFCoder will take care of everything else.

4) Selecting the **Automatic** option in the Encoder menu lets you encode the audio and video using separate encoders. For example, if you select MKV+H264+AAC then FFCoder will use x264 encoder to encode video, NeroAAC to encode audio and MKVMerge to create the MKV file. This lets you use the best features of each encoder.

5) Most encoders are **not multi-threaded**. If you have a dual core CPU then only a single core is used during encoding and the other core remains idle (50% CPU usage) . You can use FFCoder to encode 2 files simultaneously to fully utilize your CPU (and reduce encoding time by half).

6) This GUI is **very efficient**. During encoding the program uses less than 1% CPU to manage the encoding tasks and display the progress. This leaves 99% of the CPU free for use by the encoder. You can verify this using the Windows Task Manager or by using Process Explorer.

12. Known Issues

Making a single GUI for 16+ encoders is not easy. Each encoder has its own quirks and oddities. The following is a list of known issues/problems.

12.1 Issues in FFCoder

As far as the GUI is concerned, I am trying my best to find and fix as many problems as possible. Please report any bugs that you come across, and download any updates as soon as they are released.

1) **64-bit Windows:** This program has not been tested on **64-bit XP/Vista/Win7**. In most cases FFCoder should work correctly but in some cases it may fail. If the program crashes **on adding files** to the input list then try replacing MediaInfo.dll in the application folder with the 64-bit version from the MediaInfo site.

12.2 Issues related to 3rd party tools

1) **FLAC, OptimFROG and APE encoders**: Progress will not be displayed during encoding. I have been unable to capture the progress information that the encoder writes to the console during encoding. Some of these audio encoders don't support encoding from pipes. So temporary WAV files will have to be created.

2) Most encoders do not support input files with **accented (non-ascii) characters in the file name** or path. Such files will cause the encoder to crash and the encoding will fail. Only solution is to remove all accented characters from the file name and file path by renaming the file or moving it to another folder.

3) **Encoding with VFW codecs using AVS-2-AVI:** The avs2avi tool doesn't work on some windows systems. If the tool is not working then you will not get the pop-up window for selecting the VFW codec.

4) MPlayer cannot play **Speex, AMR** and **MusePack** files correctly. If you try to play such files using the in-built player then you will only hear noise even though the files produced are okay. Even **Dirac** videos don't play correctly with any MPlayer build that I tried.

5) If you try to **copy the audio or video track using Ffmpeg or MEncoder** without re-encoding then you ***may*** have the following problems (depends on your luck :-)

   - ✓ Encoding may fail
   - ✓ The audio and video may be out of sync
   - ✓ The whole file may get encoded even if you have selected only a portion of the input file for encoding (using the **Trim** right-click option).

   Avoid copying streams if you don't want to run into these issues.

6) The **bitrate options** for FFmpeg may not work correctly while encoding with the **theora** or **vorbis** codec. So for example the file produced may have an audio bitrate of about **64kbps** even though you had specified **128kbps**. This problem is not there in all FFmpeg builds. Some builds are okay.

7) The **Extract images** right-click option doesn't work with some FFmpeg builds.

FFmpeg and MEncoder are under constant development. FFmpeg for example undergoes more than a hundred revisions every week. Most of these issues may be resolved in the near future. Try to update the encoders at least once a month.

13. Feedback

In order to improve this program I need your feedback. Apart from the usual bug reports, I also need to know what you, as a user, feel about this program. What are the features that you like? What features need more improvement or can be done in a better way? What are the things that you find most irritating/confusing in this program? If you find any bugs, or if you have any suggestions, feel free to email me, leave a comment on my blog or post in the FFCoder development thread in Doom9's forums. It may not be possible for me to reply to every mail but I can assure you that your mails will not go unread.

Looking forward to hearing from you.

Regards,
Tony George

Email:
tony.george.kol@gmail.com

Discussion thread on Doom9's forums:
http://forum.doom9.org/showthread.php?t=149452