

*Instant!  
Expert!*

# SIMPLE ENCODING RECIPES FOR THE WEB 2010

MPEG-4 H.264

MPEG-4 (Simple Profile)

Windows Media

Flash 8 or 9 (On2 VP6)

YouTube

FOR TELESTREAM EPI-  
SODE PRO 6



## Forward

These Recipes do not pretend to teach you about compression and encoding. Instead they are simple recipes that will get you great results with minimal effort when you use Episode Pro.

I've also included a section on encoding for YouTube (or other video sharing sites that encodes whatever is uploaded) which is a common area of confusion.

Intelligent Assistance, Inc.

519 S. Victory Blvd

Burbank CA 91502

<http://www.intelligentassistance.com>

Copyright ©2007-2010 Philip Hodgetts and Intelligent Assistance, Inc.

All rights reserved.

# TABLE OF CONTENTS

<i>Introduction</i>	<b>5</b>
The encoding triangle	5
<i>Choosing a Format</i>	<b>6</b>
QuickTime	.mov
6	
Flash 7, 8 or 9 - .flv	6
Flash 9 release 3 (Nov 2007) and Flash 10 - .f4v or .mp4	7
Real Video	7
Windows Media 9/10	7
MPEG-1, MPEG-2 and MPEG-4	8
DivX	9
WebM (On2 VP8/MKV)	10
DRM Support	10
When not to use MPEG-4 H.264	10
What is HTML5?	11
The Best way to create MPEG-4 H.264	11
<i>Encoding Techniques</i>	<b>12</b>
Work from the source	12
Preprocessing	12
De-interlace	12
Reverse Telecine/Remove Pulldown	14
Crush (Restore) the Blacks	15
Crop encodes targeted at smaller than SD sizes	15
Gamma	16
The rule of 8	17
<i>Encoding Recipes</i>	<b>18</b>

<b><i>Workflows</i></b>	<b>19</b>
Creating and Saving a Workflow in Episode	20
<b><i>The Recipe for H.264 MPEG-4 in Episode Pro</i></b>	<b>21</b>
<b><i>Flash 8 or (early) 9 FLV</i></b>	<b>24</b>
The Recipe for Flash 8/9 in Episode Pro	26
<b><i>Windows Media</i></b>	<b>28</b>
The Recipe for Windows Media 9 in Episode Pro	30
<b><i>Working with 16:9 Source</i></b>	<b>32</b>
16:9 Letterbox	32
16:9 Anamorphic or Square Pixels (HD)	33
16:9 Letterbox Crop settings in Episode Pro from SD Source	34
16:9 Anamorphic/Square Pixel (HD) settings in Episode Pro	35
<b><i>Single Pass CBR vs multipass VBR</i></b>	<b>36</b>
<b><i>Encoding for YouTube, or other Video Sharing Site, for Maximum Quality</i></b>	<b>38</b>
Recipe for YouTube in Episode	40
<b><i>Appendix 1:</i></b>	<b>42</b>
What's the difference between a codec and a container or wrapper?	42

# INTRODUCTION

In the billing for a presentation at the Los Angeles Final Cut Pro User Group meeting about encoding, Head Cutter Michael Horton introduced the segment saying that I would be telling the group:

“How to make great looking videos for the web that are small in file size and take no expertise to create.”

Unfortunately that was an exaggeration. Encoding, the term I prefer to use over “compression,” is a skill set at least as complex as editing. Besides, if I did have the secret to great looking, low bandwidth encoding at a simple push of a button, I wouldn’t be writing it into a PDF to sell cheaply. If I had that secret, I’d be out there in the marketplace capitalizing on it. After all, the “World’s Greatest Compressionist” used to charge \$3000 a day consulting until he went to work at Microsoft.

What this book will give you are some simple recipes to produce reasonable quality video without going into the techniques that a full time encoding professional would use to get great quality. For example, everyone loves the quality of the Movie Trailers at Apple’s website. To get that quality, Apple has an employee who works from the highest quality source files and spends days to optimize the encoding for each Trailer. He encodes just a few trailers a week. To achieve that he experiments, tests, tests and tests again with each test requiring a re-encode.

You could test for optimal results and you will need to if you want to produce great looking video in absolutely the least bandwidth (i.e. the smallest file size). This book isn’t for you. This book is for those who want relatively simple formulas that will always produce a good result.

Think of it like cooking. Most cooks can follow a recipe but it takes years of experience to come up with the recipes.

## The encoding triangle

Any time we encode video we trade off three values: Quality, File Size and Ease of encoding. Like most of these type of 3-sided trade-offs you get to pick two.

- Small file size and high quality is not easy.
- High quality and easy encoding means a large file size.
- Small file size and easy encoding usually means the quality will be compromised.

However the simple recipes provided here will give good quality at reasonable file size with simple settings.

## CHOOSING A FORMAT

Much has changed in online video since the first edition of these Simple Recipes. Back then the choice of format was much less straightforward, but fortunately we now have an almost universal container and codec choice: H.264 video with AAC audio in an MP4 container.

This combination is, of course, supported in the QuickTime Player; as an HTML5 element in most browsers (other than FireFox); it is the preferred video format for Flash Players; it is a natively supported codec in Silverlight and is natively supported in Windows Media Player on Windows 7.

For a full explanation on the difference between codecs and containers check out Appendix 1: “[What’s the difference between a codec and a container or wrapper?](#)” adapted from a blog post I did at [PhilipHodgetts.com](#).

### QuickTime .mov

QuickTime as a distribution format is dead. QuickTime still fills a vital role in production, particularly within the OS X world, but has not been developed as a distribution format for many years.

Apple standardized on MP4 - both Simple Profiles (MPEG 4) and Advance Video Codec - AVC, which is also known as H.264. (The double naming comes about because it is both an MPEG standard codec, but an ITU (European) Standard as well.

Other than H.264 support there has not been a new codec for QuickTime since Sorenson Video 3, an ancient codec by modern standards. In fact, these codecs no longer display for export. That’s fine because there’s no reason to use Sorenson Video 3 (and certainly never earlier codecs) when the old MPEG-4 (Simple Profile) is of similar quality and the encoder is free to users.

### Flash 7, 8 or 9 - .flv

Flash 7 video support is bad. Unless you absolutely have to encode for very, very old Flash, there’s no need to use it. Flash 7 used the Sorenson Spark codec, which is really a slight rehash of the decades old H.263 codec. ***Not a good choice. I recommend against Flash 7 video.***



Flash 8/9, on the other hand, uses the VP6 codec from [On2 Technologies](#) and is a viable contender. Flash is installed on most computers used for the Internet and, even if they need an update, it's a small one.

Flash 8/9 was a good choice for general distribution *at the time*, but it required a 3rd party encoding tool for best quality. These encoding tools are no longer for sale by On2 but Episode continues to support On2 VP6 via the Flash 8 presets in Episode. Episode presets for Flash 9 use H.264 and MP4.

Because of the need for a proprietary codec and encoders, and because later versions of Flash play industry standard files, I **recommend against the VP6 Flash codec** but it has uses.

## Flash 9 release 3 (Nov 2007) and Flash 10 - .f4v or .mp4

With Flash 9 release 3, which was finalized in November 2007 so it is now widely spread, Adobe decided to abandon proprietary codecs and support MPEG-4 H.264 video and AAC audio – the current industry standard.

For Flash playback the MPEG-4 H.264 file must be called by the flash player instead of embedded, but it is practical to use the same file for playing in the ubiquitous Flash ecosystem as well as for Apple's iPod/iPhone/iPad and Apple TV.

***I strongly recommend using MPEG-4 H.264/AAC files with Flash*** unless you specifically need to have an alpha channel for compositing in the Flash player. If that makes no sense, use MPEG-4!

### F4V or MP4?

Adobe suggests the .F4V suffix for MPEG-4 H.264 files for use in Flash. This is unnecessary and makes it incompatible with other players. Use .mp4 for all MPEG-4 files because that is the correct suffix. Apple's wants us to use .M4V so the file will automatically open in iTunes. Adobe wants us to use .F4V so the file will open in Flash Player or Adobe Media Player. Use .MP4 for all MPEG-4 files.

## Real Video

Once the king of Internet video, Real doesn't have much of a place any more. Real Networks are more of a content company and their codecs and container format are mostly just for distributing their content. ***I cannot think of a single reason to use Reel Video in 2010.***

## Windows Media 9/10

An OK choice for delivering to corporate America if they don't have Flash 9r3 or later installed.

WMV is still a good format, but again you'll need to buy proprietary tools for encoding on a Mac. You should at least have the free player-only version of Flip4Mac installed for playing Windows Media on your Mac.

Use of Windows Media has dropped as Flash became the dominant video player, although Flash is itself being eased out due to MP4 H.264 and HTML5.

Unless specifically requested, or because a client unwisely insists on DRM (well outside the scope of this booklet) *there is no role for Windows Media* because of its proprietary codec.

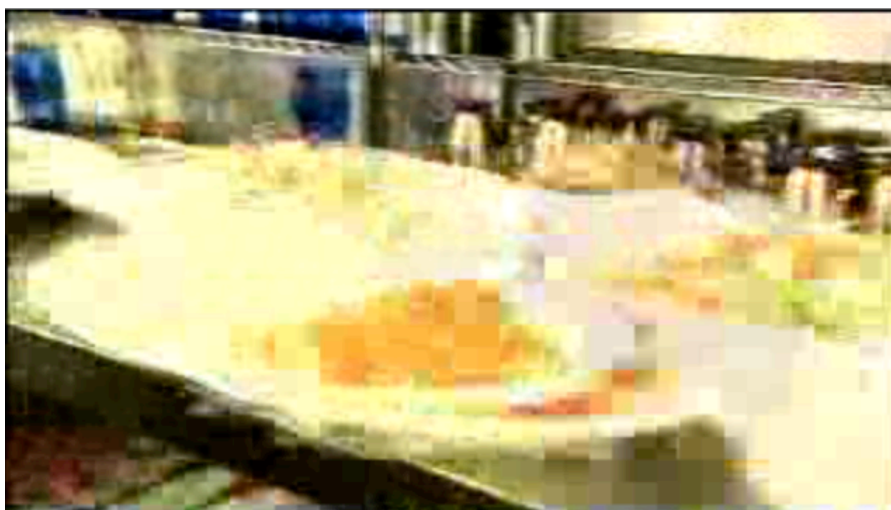
## MPEG-1, MPEG-2 and MPEG-4

Probably used to be the most ubiquitous format: every player will play MPEG-1 but it's an old codec and unless you're going to buy the Digi-gami MegaPEG Pro encoder, you're not going to get good quality with low bandwidth or ease. I think we can eliminate that from regular consideration. ***Do not use MPEG-1 any more.***

MPEG-2 is the format for DVD, Digital Television and, for now, Digital Cable and Satellite, but it's not a web format. ***Do not use MPEG-2 for web video.***

Two entries for MPEG-4 don't make it twice as good! Apple supports two variations of MPEG-4 and both use the .mp4 container.

Support for MPEG-4 was added to QuickTime at version 6.0.2 after some delay waiting for acceptable licensing conditions. QuickTime 6 supports MPEG-4 video at what is known as "Simple Profile" — up to 2 Mbit/sec.



***MPEG-4 Simple Profile***



***MPEG-4 H.264 Video from the same source at the same data rate as the Simple Profile Example.***



You would use MPEG-4 like this if you need compatibility with QuickTime 6 era players or if the target audience was using quite old, slow computers. MPEG-4 simple profile requires a lot less computing power to play than H.264.

However, I **do not recommend MPEG-4 simple profile for general web video**. The codec is old and of poor quality compared with H.264 video and AAC audio.

***I do recommend that you create an MPEG-4 with H.264 video for higher quality at lower bandwidth and always use the .mp4 suffix.***

Compare the quality between the two images on the previous page: both are from the same source (ABC TV Australia's *The Cook and the Chef* opening). Both examples are encoded at the same data rate. The difference in quality is the difference between the older Simple Profile encoder and the newer H.264 encoder. (Yes, codecs make that much difference.)

H.264 is technically MPEG-4 Part 10 – Advanced Video Codec, which is why it's sometimes known as AVC. H.264 is the name for it in the European ITU which shares the same standard. Nice thing about standards is there are so many of them!

H.264 scales really well so you can use the same codec from cell phones to digital cinema, which is unusual because most codecs are optimized for a small range of sizes, but H.264 is a great all-round codec.

***MP4 with H.264 video and AAC audio would be my choice for regular web video.*** It has one other **big advantage**. MPEG-4 both Simple Profile and H.264 are the only formats compatible with video iPods, iPhone, iPad and Apple TV, so if that's your target - or part of your target - the decision is easy.

## DivX

One last common choice. DivX is an odd case: an MPEG-4 compliant video track with an MP3 audio track in an AVI container. This is a variation on the MPEG-4 Part 2 Simple Profile video codec (the Advanced Simple Profile to be precise), which we've already discarded because of quality reasons from that older codec.

DivX got its popularity with the "pirate video" community and that's probably where it's still the strongest. QuickTime Player will play most DivX files although, if they have VBR MP3 audio, it might be silent depending on your version of QuickTime!

DivX is capable of acceptable results but is not really mainstream and I'd avoid it for client review or regular web video where it's not commonly seen. If you want to send out your project on Pirate Bay or other peer-to-peer network then DivX might be a good choice.

## WebM (On2 VP8/MKV)

When Google purchased the On2 corporation, they gained a large pool of codecs, the ‘latest and greatest’ of which is VP8. (VP6 was used for Flash video in Flash 8 and early 9 releases.) Some time after the purchase Google open sourced the codec, bundling it with the Matroska container to form WebM. There are no commercially available encoders for this new format, and playback of MKV is limited to Open Source players like VLC.

Given that VP8 is generally considered to lag H.264 in quality, other than the “open source” (but possibly not patent free) nature, there’s no compelling reason to consider WebM.

## DRM Support

I have no idea why you would want to use DRM since it does not work and only devalues the content, but if you have some insane desire to make your project hard to work with, then these are your DRM options.

**NOTE:** Setting up a DRM system is quite complex and way beyond the scope of “Simple Recipes”

Windows Media supports DRM but only on Windows.

Flash Media Server supports DRM’d media to any platform that Flash runs on.

DRM support for HTML5 video playback is in its very early stages but will be in place during 2010.

## When not to use MPEG-4 H.264

Although my standard recommendation is MPEG-4 with H.264 video and AAC audio there are some situations where you might consider using an older format or codec. This will be determined by your target audience, if it is a specific target (as compared with a general Internet audience).

If your potential viewers are inside a corporate network, I would recommend using a Flash Player to call the H.264 MPEG-4 file. Flash is installed within most corporate networks. If it’s not installed on your target network, then Windows Media would be a fallback position.

Otherwise I’d tend to encode to MPEG-4 with H.264, or a .mov with H.264 if you want to use other QuickTime features like a chapter track.

## WHAT IS HTML5?

“HTML5” is a convenient shorthand for a combination of technologies that together provide the framework for Rich Internet Activities, that otherwise would require Adobe Flash. HTML5 itself is the fifth iteration of the HTML standards. This version is interesting because it provides for playback of video without the need for a plug-in like the QuickTime Player or Flash Player by use of the <video> tag.

Nothing in life is simple and, while H.264 MP4 is supported for the HTML5 <video> tag in Safari, Chrome and some other minor browsers, Firefox’s Open Source creators aren’t happy with a commercially licensed codec and instead are pushing for the Ogg Theora and Vorbis codecs for video and audio in the HTML5 <video> tag. Google entered the fray with its purchase of On2 and has released the On2 VP8 codec as Open Source in a MKV package, calling the result WebM.

For our purposes, we can consider H264 MP4 - our standard encode - as being sufficient for HTML5’s <video> element. Browsers that don’t support it directly will usually “fall through” to supporting the file in a plug-in. (Best practice in web design mandates that fall-through process but it’s not an encoding issue.)

While not related to encoding video, HTML5 also uses the <canvas> tag and CSS for animation and Javascript for interactivity.

## THE BEST WAY TO CREATE MPEG-4 H.264

To encode an MPEG-4 H.264 file in Episode Pro, choose one of the iPod or Apple TV specific settings as a starting point. Encoding for iPod seems to require some “secret sauce.”

Starting with an iPod compatible setting ensures playback on Apple devices and Flash compatibility. iPod settings work on iTouch and iPhone. Avoid targeting these players exclusively as the files will be smaller than recommended, at lower quality than we want.

For Apple TV, use the “Export for Apple TV” setting.

**Note:** Apple TV settings are not compatible with some iPods and iPhones when encoded from HD source.

**IMPORTANT:** Videos created by these presets have a .m4v file suffix instead of .mp4. There's no difference between the two except that double-clicking a .mp4 opens the file in QuickTime Player while double-clicking a .m4v opens the file in iTunes. For web use, change the file suffix from .m4v to .mp4 since many web servers don't know how to treat a .m4v file.

# ENCODING TECHNIQUES

## Work from the source

Always work from the source. Encoding already encoded video is not going to deliver good results because the encoding artifacts from the first encoder are going to be preserved by the second encoder, because it thinks the artifacts are details in the image, while it introduces its own artifacts. So, always work from the source, preferably uncompressed video or at least the native format of the edit.

Never use a video that has been encoded to Sorenson Video, Cinepak, MPEG-4, H.264 or other distribution codec. DV and HDV are acceptable if they are the native format. Also acceptable, if no real master is available, is a “rip” from a DVD using MPEG Streamclip, DVDxDV or Cinematize.

## Preprocessing

A very large part of what a professional encoding specialist will do is in the pre-processing of the video before it's encoded to make it easier for the encoding software to get the best possible image.

Pre-processing is out of the scope of “Simple” recipes for encoding but it should not be overlooked as you develop skills in this area. The most bang-for-buck pre-processing is to:

- De-interlace;

- Crush the blacks in the image; and

- Crop (if reducing the image size below 640 x 480 or 640 x 360)

## De-interlace

NTSC and PAL video are interlaced. Much HD source is interlaced as well. All interlaced source should be de-interlaced for web video because all computer screens are progressive. Therefore video for the web must be de-interlaced.

**Important Note:** If the source is progressive DO NOT de-interlace

You need to know the software you're using and work out how it goes from interlaced 480/576 lines to 640 x 480 or 640 x 360 lines. I prefer to de-interlace first, then allow the software to scale the image.

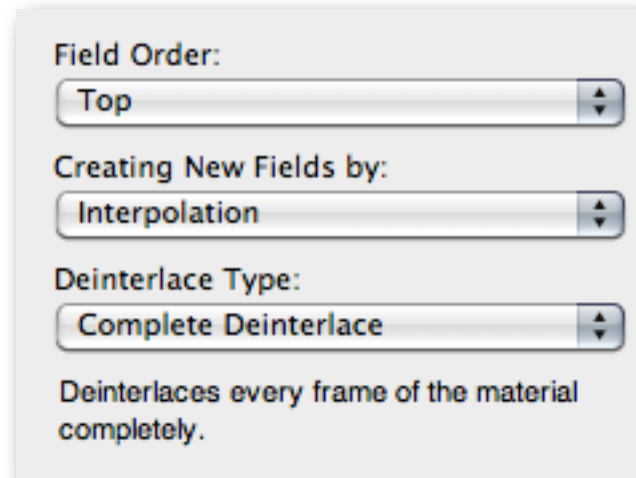
## Simple Deinterlacing (for encodes 320 x 240 or 320 x 180 ONLY)

Most encoding tools have a de-interlace control. In Episode Pro they're in the Video Filters area.

For small web video such as 320 x 240, the simplest de-interlace is preferred because it is the fastest and because we're not needing to see the results of two fields. These days I consider 320 x 240 to be too small and recommend 640 x 480 or 640 x 360, which requires Advanced De-interlace. (See next section.)

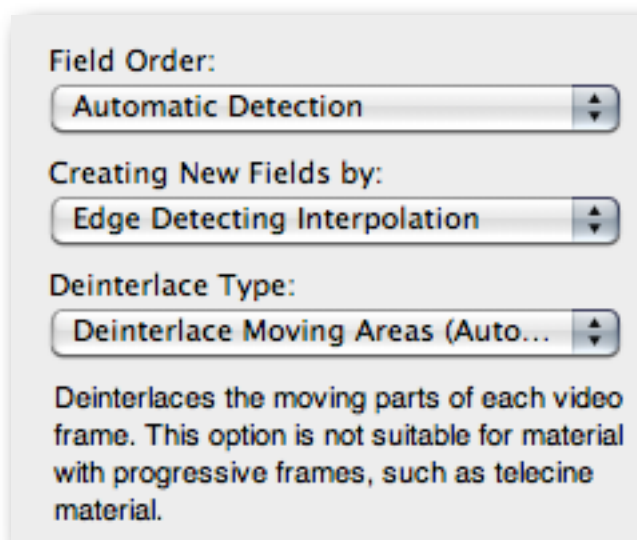
In Episode Pro, match the settings to the right:

Simple deinterlacing like this is really only suitable for sizes below 320 x 240 (one field height).



## Advanced Deinterlace (the default)

Since we actually recommend a 640 x 480 or 640 x 360 encode for general web video and Apple devices, a more sophisticated de-interlace is required to keep the frame size large (effectively SD size translated to square pixels) and retain as much detail from the source as possible. An adaptive de-interlace will only de-interlace areas that are in motion, and will synthesize the moving part by interpolation (i.e. making new images to fill what's been taken out).



Episode Pro, using the settings to the left, will detect which parts of the image are moving and then de-interlace those parts of the image and interpolate (create new pixels between the values either side) only in the moving parts. This retains maximum quality but is much slower to encode.

For a faster encode, select 'Complete Deinterlace' for 'Deinterlace Type, which will deinterlace the entire image, even those parts that do not require deinterlacing.

## Reverse Telecine/Remove Pulldown

If you have a source file that has 3:2 (Regular Pulldown) on a file that was shot as 24P you must remove the pulldown before encoding and only encode the progressive frames. It is better to obtain the 24P master before pulldown was added to avoid the necessity for this step.

**Note:** Episode Pro uses the Frame Rate tool to Inverse Telecine. Inverse and Reverse Telecine have the same meaning.

The most generally appropriate setting for Reverse Telecine is to use these settings:

Filter Mode:  
Fixed Frame Rate

Framerate Preset:  
Inverse Telecine 29.97->23.976

New Framerate (Frames Per Second):  
23.976

Change Framerate to:  
1/2

First Interlaced Pair:  
Frame 2 and 3

However, if you know for sure that the pulldown pattern is consistent you can speed the encode by telling Episode that there is a fixed cadence (pattern to the pulldown) and specify which frames are the split frames.

Filter Mode:  
Fixed Frame Rate

Framerate Preset:  
Inverse Telecine, Fixed Cadence...

New Framerate (Frames Per Second):  
23.976

Change Framerate to:  
1/2

First Interlaced Pair:  
Frame 2 and 3

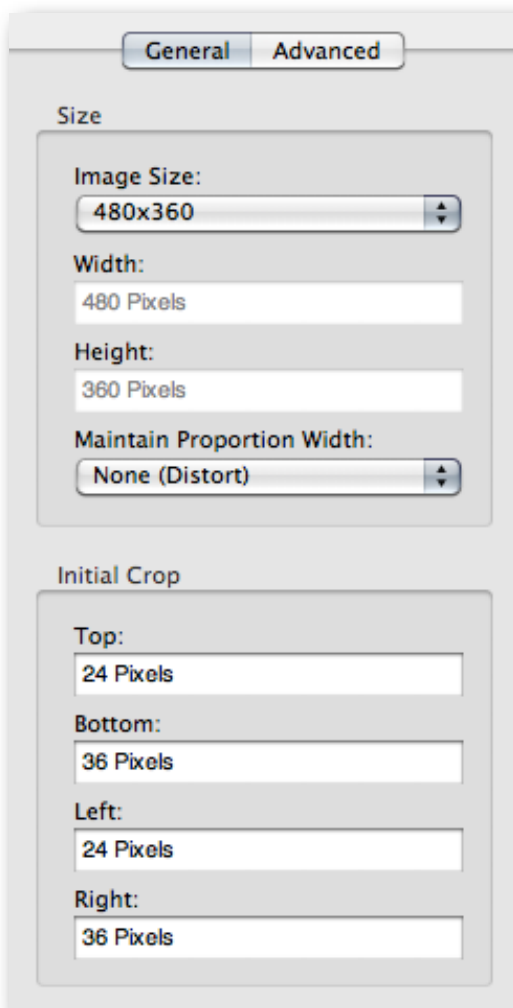
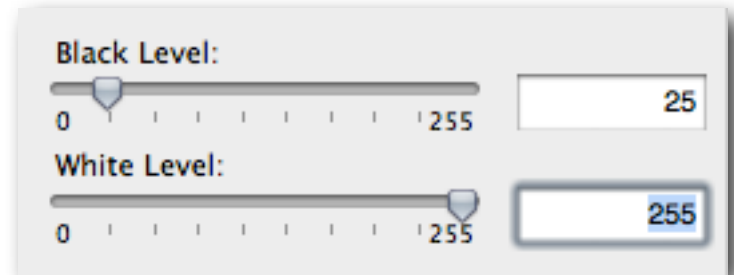
## Crush (Restore) the Blacks

Black is complicated in video. As the signal gets closer to black, it becomes noisy. Noise – a random dancing pattern – is a problem for encoding software because it does not know that noise is unimportant, so it devotes large amounts of the available data rate to preserve the noise. This is exactly what we do not want!

So we want to make all the near-black values into real black. To do this we use a tool called Black Restore in most encoding software.

In Episode Pro set the Black level to its value from within the 0-255 range. Usually something in the 20-30 range will kill the noise in the very darkest parts of the image without noticeably darkening the image.

This simple technique probably brings the greatest benefit to images that are dark or have dark areas, and still need to be encoded.



## Crop encodes targeted at smaller than SD sizes

In traditional video, at least 5% of the image from each side was never intended to be seen. This is the “Safe Action” zone and was an allowance for the part of a CRT that was hidden behind the cowling. Of course, in web video we see from edge to edge.

Since this part of the image was never intended to be seen, it rarely has anything essential to keep in so we can crop it away when we are encoding to a finished size smaller than 640 x 480.

**NOTE:** Do not crop when going to 640 x 480 or 640 x 360 (16:9) or for HD sizes, because that would require the encoding engine to scale the image larger, which would reduce quality.

Also note that we do not crop HD sizes before scaling because they are designed without significant overscan – HD displays are usually LCD or Plasma and (mostly) display the full raster.

Five percent of 720 pixels is 36 pixels.

Five percent of 480 pixels is 24 pixels. Five percent of 486 pixels is 24.3 pixels, so for both 480 and 486 (NTSC) we use 24 pixels.

In PAL territories, the same horizontal measure applies, but 5% of 576 pixels is 28.8 pixels. Given that we like whole numbers (in fact most cropping tools only allow whole numbers) you'd think can go with 28 or 29, but in practice only 28 is appropriate because we work (mostly) in even numbers.



**Figure 1:** With low action sources, there is no noticeable difference between the single and multi-pass. This is because this footage has fairly constant amount of low action so allocating bits from parts with lower motion to parts with higher motion gives no pay-back.

To view this movie in a browser, click [here](#).

## Gamma



**Figure 2:** After applying the recommended gamma and crop settings, the Gorilla footage looks like this;

The image has better grayscale values, noticeable around Ethan's head.



## The rule of 8

A good rule of thumb is to make sure that any dimension we work with a codec is divisible by 8. Exactly divisible by 8. Codecs work most efficiently when they are working in blocks of 16, 8 or 4, with the most common requirement being blocks 8 pixels square. If we were to throw in an odd number, like cropping 29 from top and bottom, the codec has to pad out to the next largest block of 8 pixels, even though most of the pixels in the block will be without content they still take up a finite bandwidth.

On decode the codec has to eliminate the redundant parts before it displays, reducing the efficiency of the codec and making both encoding workstation and playback computer work harder.

**Simple Rule: Always make sure both dimensions of your encoded video are divisible by 8.**

We use the same rule here, so that the scaling will happen more linearly than if we did a different crop value, keeping maximum resolution.

## ENCODING RECIPES

These recipes are designed for most types of footage. They are not optimized for “easy” or “hard” to encode material. Source with lots of cuts, lots of camera moves and lots of action are more difficult to encode than talking heads so we allow for the worst-case scenario. If your source is primarily talking heads, you can reduce the data rate significantly.

For these tests I used two pieces of source material: a talking head of Ethan Markowitz demonstrating the [Gorilla](#) software at a Los Angeles Final Cut Pro User Group meeting (Gorilla) and a short piece of mountain bike footage (Bikes). The Gorilla footage is relatively easy to encode; the Bikes footage is more challenging because of the rapid motion.

There are encoding recipes for:

- MPEG-4 H.264 with AAC audio. This is the recommended file format for most purposes. It will work with QuickTime Player, MP4 players, in web browsers, for Flash, HTML5-capable browsers, Windows Media Player on Windows 7, Silverlight and for Apple Devices.
- Flash 8 a.k.a. On2 VP6. For targeting older Flash installations where it's not possible to update old Flash 9r3 (or earlier) installations. Note that Flash support for H.264 MP4 was fully released by November 2007, so there should be few situations that require Flash 8 compatibility.
- Windows Media 9/10 for those situations where Flash is not installed in corporate networks. Windows Media only works in browsers with the Windows Media or Flip4Mac (OS X) plug-in.

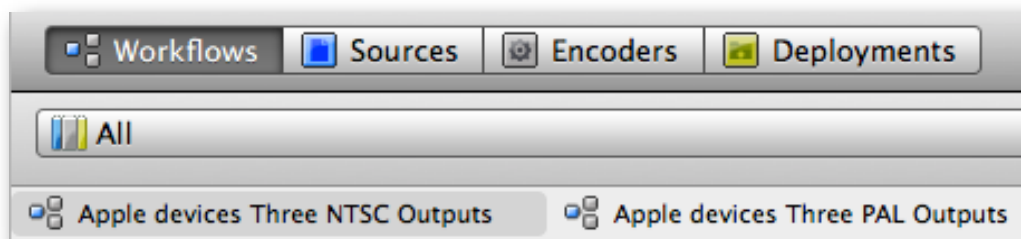
The actual movies are available online (from a link near the still) so you can see the movie in motion.

For comparison of the difference between the default of Multi-pass (the default, but slower) and single pass, see the Section on [CBR vs VBR](#).

# WORKFLOWS

Episode 6 introduces Workflows-based settings so you can either start with a complete workflow or build your own. Conveniently the different sizes we'd most commonly need are built into the one workflow. Drag the NTSC or PAL set to the Workflow area of the interface.

This gives you a workflow that will take one source and encode it to the three most-commonly needed sizes as shown to the right.:



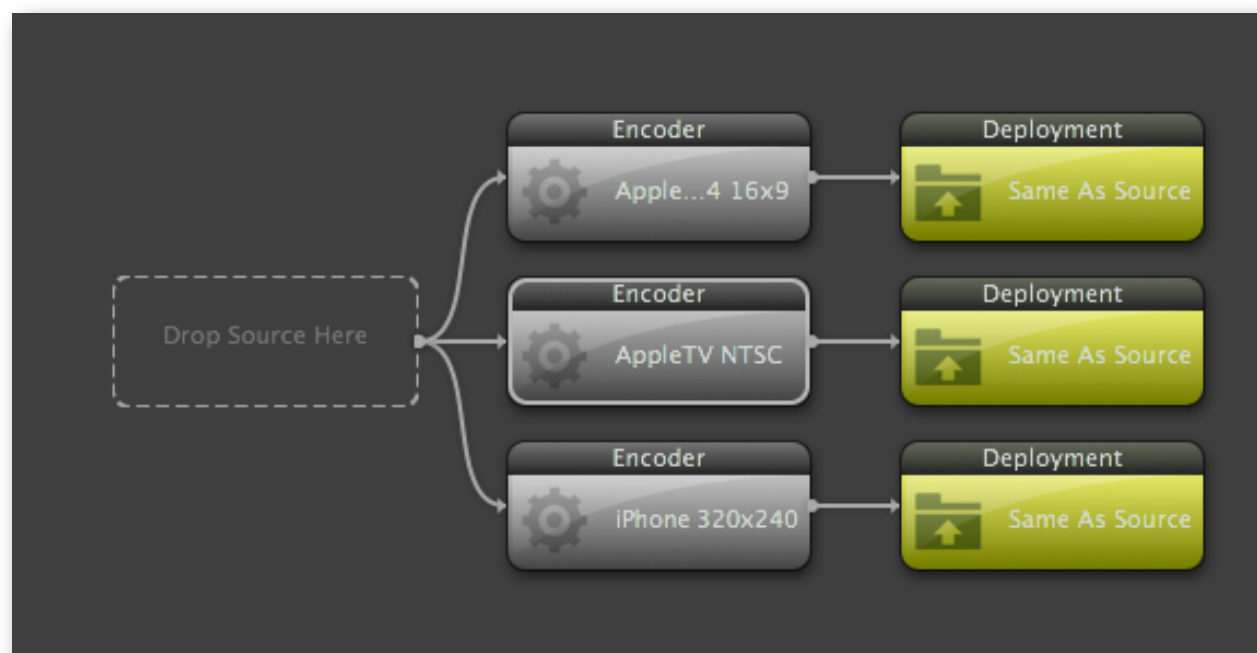
Of course, an Apple TV version only makes sense if you have an HD source!

With the three workflows in the workflow area, you can simply delete the ones that you don't want, and save a new workflow. Or leave the workflow like it is and have three versions for different purposes.



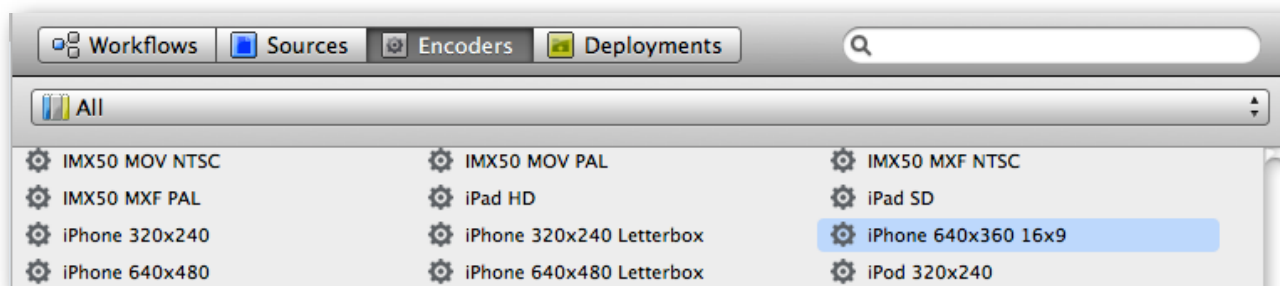
The most common size you'll create is a 640 x 480 (4:3) or 640 x 360 (16:9)

whether from SD or HD material, so you can delete the unwanted encodes from this set by selecting the Encoder module and deleting it. However, it's better to make your own Workflow and it's easy.



## Creating and Saving a Workflow in Episode

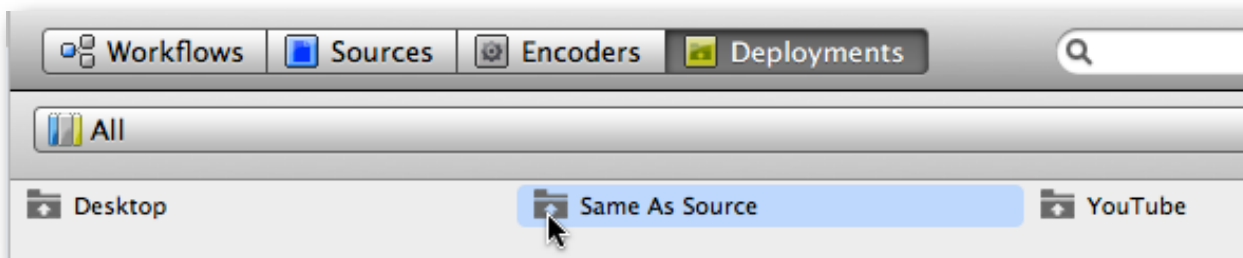
1. Save any previous workflow if it is not saved, and you may want to re-use it.
2. Clear the Workflow area by clicking on the 'Clear' button at bottom left of the Workflow display. Acknowledge that you want to clear the Workflow area.
3. Choose the Encoders tab and locate the Encoder you plan on using.



4. Drag the Encoder to the Encoder slot in the Workflow area.



5. Select the Deployments tab and locate the Deployment you want to use, or create a new Deployment.



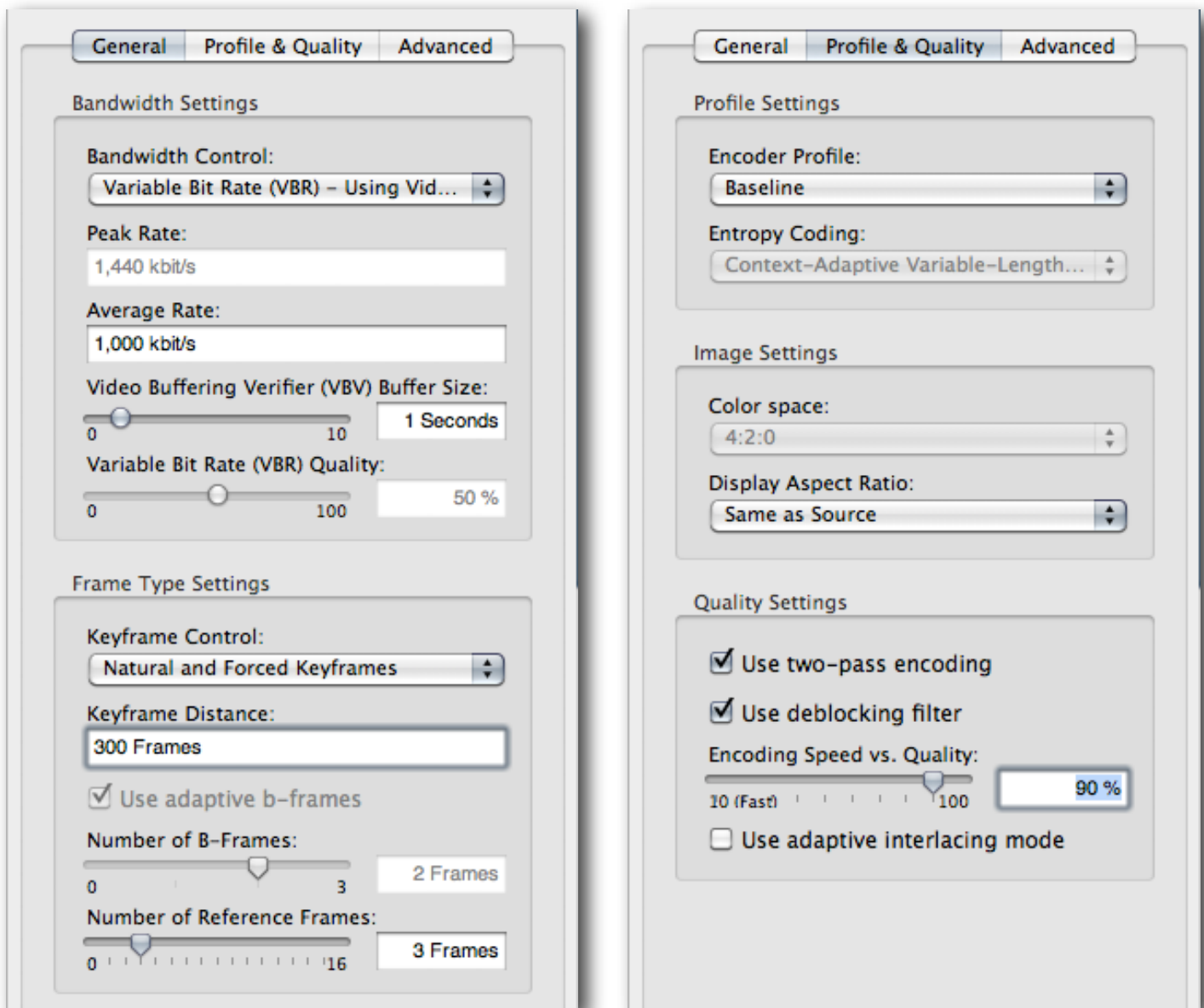
6. Drag the deployment down to the Deployment position to the right of the Encoder.



7. Name and save the Workflow for future use.

# THE RECIPE FOR H.264 MPEG-4 IN EPISODE PRO

In Episode Pro start with the iPod\_H264\_640 x 360 preset (or 640 x 480 for 4:3 source) if you want iPod compatibility. I'd recommend the iPod one as starting point because it's closest to what we want.



**Tip:** If people are unlikely to search back or forward, you can increase the efficiency of the encode by selecting Natural Keyframes Only for Keyframe Control.

You'll need to add (if not already there) a Deinterlace Filter (if the source is interlaced), Black and White Restoration and a Resize filter.

## Deinterlace

Field Order:  
 Automatic Detection

Creating New Fields by:  
 Edge Detecting Interpolation

Deinterlace Type:  
 Deinterlace Moving Areas (Automatic)

Deinterlaces the moving parts of each video frame. This option is not suitable for material with progressive frames, such as telecine material.

Double frame rate

Threshold:  
 0  230

Deinterlace chroma

## Black and White Restoration

Black Level:  
 0  255

White Level:  
 0  255

## Resize

General Advanced

Size

Image Size:  
 Custom...

Width:  
 640 Pixels

Height:  
 360 Pixels

Maintain Proportion Width:  
 None (Distort)

Initial Crop

Top:  
 0 Pixels

Bottom:  
 0 Pixels

Left:  
 0 Pixels

Right:  
 0 Pixels

General Advanced

Scaling Details

Interpolation Method:  
 Automatic

Bicubic Sharpening:  
 0

Preprocessing:  
 Nothing

Interlace Handling:  
 Progressive Output

Aspect Ratio

Input Display Aspect Ratio:  
 Derive From Source

Output Display Aspect Ratio:  
 16:9

Output Pixel Aspect Ratio (MOV only):  
 Square

Horizontal Pixel Spacing:  
 1

Vertical Pixel Spacing:  
 1



**Figure 3:** A fast-moving frame shows up compression artefacts.

The H.264 MPEG-4 version looks the best of all, at matching data rates. This is the 2-Pass VBR version.

To view this movie in a browser, click [here](#).

**Figure 4:** Slow moving clips show no benefit from 2-Pass VBR.

To view this movie in a browser, click [here](#).



## FLASH 8 OR (EARLY) 9 FLV

Flash 8/early 9 uses the On2 VP6 Codec. This codec is a big step up from Flash 7 but it is still not as good as WMV, H.264 or MPEG-4. You may need to use this format if you're targeting older installations in corporate America. The Version 9 release 3 of Flash supports MP4 H.264 (which is what I recommend) and that is now the most commonly installed version of Flash, with around 90% of browser plug-ins supporting it.



**Figure 5:** FLV Encode from Episode to On2 VP6 Flash Video using 1 Pass CBR. This is at the same 1000 Kbits/sec, crop and gamma settings as for all the other encodes. This **appears** to be lower quality than the other formats.

To view this movie in a browser, click [here](#).

This quality is comparable with WMV or MPEG-4 H.264, as one would expect with the On2 VP6 codec, **when viewed from a SWF player on a website**. The quality is significantly better than when playing the FLV locally.

**IMPORTANT:** On a Mac you cannot tell the final quality on the Internet, by viewing the FLV file in a Mac-based player. Upload to a server and test quality from there.





**Figure 6:** FLV Encode from Episode Pro to On2 VP6 Flash Video using 2 Pass VBR. This is at the same 1000 Kbits/sec, crop and gamma settings as for all the other encodes.

Compare this image with the same frame once embedded in a SWF player and uploaded. The embedded version looks dramatically better.

To view the uploaded movie in a browser, click [here](#).

## The Recipe for Flash 8/9 in Episode Pro

To encode to Flash in Episode Pro, you will need the appropriate version.

**Bandwidth Settings**

Peak Rate:  
1,100 kbit/s

Average Rate:  
1,000 kbit/s

Frame Skip Probability:  
0.0 | 1.0 **0.1**

Video Buffering Verifier (VBR) Buffer Size:  
0 | 60 **5 Seconds**

Variable Bit Rate (VBR) Strength:  
0 | 100 **50 %**

**Keyframe Settings**

Keyframe Control:  
Natural Keyframes Only

Keyframe Distance:  
200 Frames

Minimum Distance:  
1 Frames

**Profile Settings**

Profile:  
Vp6-E

Error resilient mode

Encode alpha

**Encoding Settings**

Complexity:  
Normal Quality

Use two-pass encoding

Bitrate:  
Constant Bit Rate (CBR)

Sharpness:  
0 | 10 **5**

Black Level:  
0 | 255 **15**

White Level:  
0 | 255 **255**

Gamma:  
-100 | 0 | 100 **20 %**

Field Order:  
Automatic Detection

Creating New Fields by:  
Edge Detecting Interpolation

Deinterlace Type:  
Deinterlace Moving Areas (Automatic)

Deinterlaces the moving parts of each video frame. This option is not suitable for material with progressive frames, such as telecine material.

Double frame rate

Threshold:  
0 | 230 **6**

Deinterlace chroma

General Advanced

Size

Image Size:  
Custom...

Width:  
640 Pixels

Height:  
360 Pixels

Maintain Proportion Width:  
None (Distort)

Initial Crop

Top:  
0 Pixels

Bottom:  
0 Pixels

Left:  
0 Pixels

Right:  
0 Pixels

General Advanced

Scaling Details

Interpolation Method:  
Automatic

Bicubic Sharpening:  
0

Preprocessing:  
Nothing

Interlace Handling:  
Progressive Output

Aspect Ratio

Input Display Aspect Ratio:  
Derive From Source

Output Display Aspect Ratio:  
16:9

Output Pixel Aspect Ratio (MOV only):  
Square

Horizontal Pixel Spacing:  
1

Vertical Pixel Spacing:  
1

# WINDOWS MEDIA

Encoding Windows Media is not natively supported on OS X but there are at least 2 solutions from Telestream:

QuickTime Export Components from Flip4Mac that integrate in any QuickTime application including Compressor or Export with QuickTime Conversion from Final Cut Pro;  
and  
Episode Pro.



**Figure 7:** Export to WMV 9 from Episode using 1 Pass CBR.

To view this movie in a browser, click [here](#). This file will play in QuickTime Player if the free Flip4Mac Windows Media [Playback](#) option is installed.

In these examples the difference between 1 Pass CBR and 2 Pass VBR is not as pronounced as the difference between 1 Pass CBR and 2 Pass VBR for MPEG-4 H.264.



**Figure 8:** Export to WMV 9 from Episode Pro with 2 Pass VBR.

To view this movie in a browser, click [here](#). This file will play in QuickTime Player if the free Flip4Mac Windows Media [Playback](#) option is installed.

## The Recipe for Windows Media 9 in Episode Pro

Start with WMV 9 High Quality from the By Format > Windows Media group.

**Bandwidth Settings**

Coding Mode:  
Two-Pass Constant Bit Rate (CBR)

Peak Rate:  
1,600 kbit/s

Average Rate:  
1,000 kbit/s

Video Buffering Verifier (VBV) Buffer Size:  
0 60 5 Seconds

Picture Quality:  
0 100 80 %

**Keyframe Settings**

Keyframe Control:  
Natural Keyframes Only

Keyframe Distance:  
4 Seconds

Number of B-Frames:  
2

**Profile Settings**

Profile:  
Main Profile

**Encoding Settings**

Encoding Complexity:  
Offline Normal

**Black Level:**  
0 255 15

**White Level:**  
0 255 255

**Gamma:**  
-100 0 100 20 %

**Field Order:**  
Automatic Detection

Creating New Fields by:  
Edge Detecting Interpolation

Deinterlace Type:  
Deinterlace Moving Areas (Automatic)

Deinterlaces the moving parts of each video frame. This option is not suitable for material with progressive frames, such as telecine material.

Double frame rate

Threshold:  
0 230 6

Deinterlace chroma

General Advanced

Size

Image Size:  
Custom...

Width:  
640 Pixels

Height:  
360 Pixels

Maintain Proportion Width:  
None (Distort)

Initial Crop

Top:  
0 Pixels

Bottom:  
0 Pixels

Left:  
0 Pixels

Right:  
0 Pixels

General Advanced

Scaling Details

Interpolation Method:  
Automatic

Bicubic Sharpening:  
0

Preprocessing:  
Nothing

Interlace Handling:  
Progressive Output

Aspect Ratio

Input Display Aspect Ratio:  
Derive From Source

Output Display Aspect Ratio:  
16:9

Output Pixel Aspect Ratio (MOV only):  
Square

Horizontal Pixel Spacing:  
1

Vertical Pixel Spacing:  
1

## WORKING WITH 16:9 SOURCE

There are two types of 16:9 source: Letterbox and full height/full frame. Given the changing nature of video source, we've assumed 16:9 source throughout, with the 640 x 360 recommendation.

With full height there are two types of source: square pixel, usually used for High Definition source, and anamorphic. All Standard Definition 16:9 is anamorphic. It has the same number of pixels as regular 4:3 but the pixels are stretched wider to fill the full screen. Both HD and SD Anamorphic are handled same way when encoding.

### 16:9 Letterbox

There is no point encoding black for the web. Although pure black takes relatively few bits to encode, why waste even one bit for irrelevant data? An iPod or Apple TV will automatically letterbox when playing back, and all other PC/Mac playback will show the full frame without black letterboxing.

So, for 16:9 Letterbox source we need to crop the black from the image and encode only the image component, telling our encoding tool that it should be displayed as 16:9.



Letterboxed 16:9 needs to be cropped to the image area for encoding. Image courtesy [Yogi Marlon](#).



Only the crop and size settings change. All other components to the recipe – data rate, gamma, etc. – are the same as for the other recipes in this book.

## 16:9 Anamorphic or Square Pixels (HD)



Full Height Anamorphic  
16:9 needs to be  
stretched to the original  
image aspect ratio before  
encoding.

Image courtesy [Yogi Marlon](#) and [ProfessorIT](#).

When the source is full height Anamorphic or square pixels (HD) then we only need to tell the encoding tool that the output should be 16:9. When displayed as 4:3, Anamorphic material will be incorrectly displayed in a regular 4:3 space, which will look tall and thin.

Square pixel images display the image aspect ratio correctly.

Image courtesy [Yogi Marlon](#) and [ProfessorIT](#).



## 16:9 Letterbox Crop settings in Episode Pro from SD Source

**NTSC**

General Advanced

Size

Image Size: Custom...

Width: 640 Pixels

Height: 360 Pixels

Maintain Proportion Width: None (Distort)

Initial Crop

Top: 60 Pixels

Bottom: 60 Pixels

Left: 0 Pixels

Right: 0 Pixels

**PAL**

General Advanced

Size

Image Size: Custom...

Width: 640 Pixels

Height: 360 Pixels

Maintain Proportion Width: None (Distort)

Initial Crop

Top: 108 Pixels

Bottom: 108 Pixels

Left: 0 Pixels

Right: 0 Pixels

## 16:9 Anamorphic/Square Pixel (SD) settings in Episode Pro

The screenshot shows the 'General' tab of the Episode Pro settings. Under the 'Size' section, the 'Image Size' is set to 'Custom...'. The 'Width' is 640 Pixels and the 'Height' is 360 Pixels. The 'Maintain Proportion Width' is set to 'None (Distort)'. Under the 'Initial Crop' section, the 'Top', 'Bottom', 'Left', and 'Right' values are all set to 0 Pixels.

## 16:9 Anamorphic/Square Pixel (HD) settings in Episode Pro

When encoding from 1080i or 1080p through Episode Pro, click on the 'Lowpass source for large downscales' checkbox for better quality images. This prevents details from the larger image being scaled down to be over-sharp.

The screenshot shows the 'General' tab of the Episode Pro settings. Under the 'Size' section, the 'Image Size' is set to 'Custom...'. The 'Width' is 1,280 Pixels and the 'Height' is 720 Pixels. The 'Maintain Proportion Width' is set to 'None (Distort)'. Under the 'Initial Crop' section, the 'Top', 'Bottom', 'Left', and 'Right' values are all set to 0 Pixels.

## SINGLE PASS CBR VS MULTIPASS VBR



**Figure 9:** Single Pass “H.264 for iPod” at default settings as per those on page 15, except for the Option of Multi-pass being turned off.

If you ever wondered whether the extra time it takes for Multi-Pass encoding was worthwhile, compare with the next example.

To view this movie in a browser, click [here](#).



**Figure 10:** Multi-pass H.264 at default settings as per those on page 20.

This example is smoother, less blocky and shows better detail.

To view this movie in a browser, click [here](#).

In motion, it's hard to tell the difference between the two examples.

# ENCODING FOR YOUTUBE, OR OTHER VIDEO SHARING SITE, FOR MAXIMUM QUALITY

***What people tend to forget is that you are sending YouTube a master for them to compress, therefore send the highest quality you can, as long as it's less than 2 GBs and 15 minutes.***

YouTube.com is well known for being the busiest video sharing site, which for the longest time used the much older Sorenson Spark codec for their video encoding. Fortunately now they support higher quality, H.264 and even “HD” right on up to “4K”. (In practice 1280 x 720 is the “sweet spot” for HD on the web at this time.)

Many people send YouTube an already compressed video, and are disappointed when they see the quality that results on YouTube. That's because most of the information was first thrown away by the encode before upload, long before YouTube started encoding.

**The goal is to give YouTube a master that they can use for encoding.**

YouTube have two limitations: no more than 15 minutes per video and no larger than 2 GB per video. Now that the file size limitation has gone so high, it's no longer a challenge to get high quality 15 minute videos to YouTube, even for HD.

YouTube converts everything that is uploaded to H.264 (for Apple devices, accessible via Click2Flash) and to the older codec for the “standard” quality.

YouTube also supports 16:9 natively now, so send in 16:9 masters instead of needing to letterbox it.

If you have HD source you should send an HD version of it to YouTube. You will need to send at least 1280 x 720 if you want the HD viewing option to become available. Otherwise send an SD version — 640 x 360 for 16:9 or 640 x 480 for 4:3.

YouTube prefers — in order — MPEG-2, MPEG-4 or QuickTime movie. Remember, you're sending a master for re-encoding so the bitrate will need to be much higher than regular distribution.

Use any application that exports to .mp4 with H.264 video, including QuickTime Player Pro, Final Cut Pro, Sony Vegas, etc. I prefer MPEG-4 with H.264 video for maximum efficiency.

- Export as MPEG-4 with H.264 and set the size to 640 x 480 or 640 x 360 for SD source; 1280 x 720 for HD source. You get to control de-interlacing and scaling.

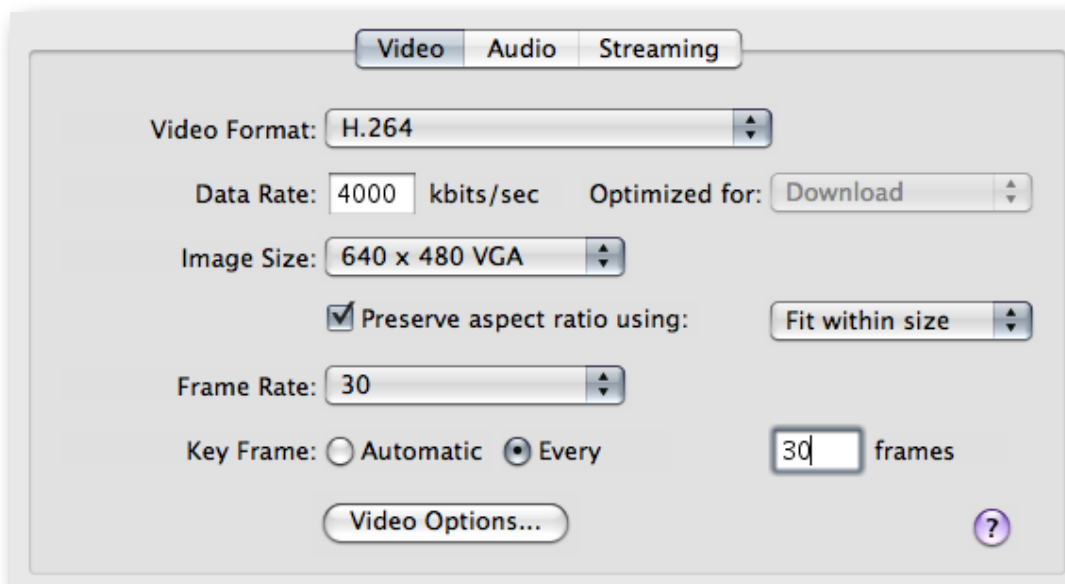
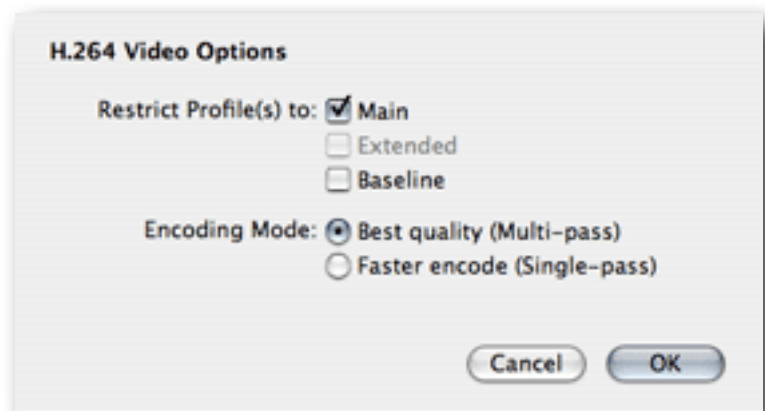
The data rate for 640 x 480 or 640 x 360 should be at least 3000 Kbits/sec. You could use less, and if you were worried about bandwidth or upload size (i.e. the time to upload) then you drop as low as 2000 Kbits/sec. As we've seen in the rest of the book, for regular distribution we're encoding these sizes to 850 - 1000 Mbit/sec.

For 720P source encode H.264 at 4-6 Mbits/sec for optimal quality. Distribution 720P would be closer to 1.5 Mbits/sec.

A simple way to ensure that the quality is there is to export to MPEG-4 with:

- H.264 video at 4000 Kbits/sec (4 Mbits/sec or 500 KBytes/sec) for SD, 6000 Kbit/sec for 1280 x 720 HD.
- 640 x 480 or 640 x 360 (from SD) or 1280 x 720 (HD)
- Stereo audio with AAC at 64 Kbit/sec per channel (or 128 Kbits/sec for stereo)
- Recommended Sample Rate of 480 KHz,
- Native Frame Rate (23.98 or 29.97 or 25)
- Best Encoding Quality.
- In the Video Options Main Profile should be checked on and Best Quality (Multi-pass) is advised for best quality.

These are the settings for QuickTime Player.



## Recipe for YouTube in Episode

Episode 6 has Workflows for both HD and SD uploads to YouTube, which deploy straight to YouTube after encoding (once you enter your YouTube credentials). By default the HD setting uses a data rate of 4800KBits/sec and SD at 1600 Kbits/sec. If the upload size is kept below 2 GB I would increase the settings to these for the highest quality encode at YouTube:

### YouTube HD (1280 x 720)

**General** | Profile & Quality | Advanced

**Bandwidth Settings**

Bandwidth Control:  
Variable Bit Rate (VBR) - Using Vid...

Peak Rate:  
5,760 kbit/s

Average Rate:  
4,800 kbit/s

Video Buffering Verifier (VBV) Buffer Size:  
0 10 5 Seconds

Variable Bit Rate (VBR) Quality:  
0 100 50 %

**Frame Type Settings**

Keyframe Control:  
Natural and Forced Keyframes

Keyframe Distance:  
100 Frames

Use adaptive b-frames

Number of B-Frames:  
0 1 3 1 Frames

Number of Reference Frames:  
0 16 5 Frames

Audio Codec AAC Quicktime

Audio streaming

Video streaming

In/Out Points

Timecode

Intro & Outro

Bit Rate:  
192 kbit/s

▶ Video Filters 368

▼ Audio Filters 369

Channel Mapper

Channels

High Pass / Low Pass

Sample Rate

Audio Speed

Offset

Fade

Balance

Equalizer

Volume

Sample Rate:  
44100 Hz



## YouTube 2D (640 x 360)

General Profile & Quality Advanced

Bandwidth Settings

Bandwidth Control:  
Variable Bit Rate (VBR) – Using Vid...

Peak Rate:  
2,160 kbit/s

Average Rate:  
1,600 kbit/s

Video Buffering Verifier (VBV) Buffer Size:  
0 10 5 Seconds

Variable Bit Rate (VBR) Quality:  
0 100 50 %

Frame Type Settings

Keyframe Control:  
Natural and Forced Keyframes

Keyframe Distance:  
100 Frames

Use adaptive b-frames

Number of B-Frames:  
0 3 1 Frames

Number of Reference Frames:  
0 16 5 Frames

Audio Codec AAC Quicktime

Audio streaming

Video streaming

In/Out Points

Timecode

Intro & Outro

Bit Rate:  
192 kbit/s

▶ Video Filters 88

▼ Audio Filters 89

Channel Mapper

Channels

High Pass / Low Pass

Sample Rate

Audio Speed

Offset

Fade


Balance

Equalizer

Volume

Sample Rate:  
44100 Hz

## YouTube Credentials

 **YouTube**  
User Created Task

Task Name:  
YouTube

Username:  
BigBrains4Rent

Password:  
••••••••

Video Info: Title:  
prEdit Demo

Video Info: Description:  
Demonstration of the prEdit workflow.

Video Info: Category:  
Film & Animation

Video Info: Keywords:  
software demo assisted editing

Private video

# APPENDIX 1:

## WHAT'S THE DIFFERENCE BETWEEN A CODEC AND A CONTAINER OR WRAPPER?

It's a subject with widespread confusion often leading to only a partial understanding.

There are file containers, sometimes called wrappers, that wrap around a number of video and audio tracks. Each of those tracks will have an appropriate video or audio codec. A codec is a concatenation of "coder/decoder". Basically it's like using a secret code or cryptography: as long as the encoder and the decoder understand each other, we get video and audio back out at the other end.

Think of a shipping container. There's this standard "wrapper" (the container) which tells us nothing. Inside could be a car, dozens of computers or a million wrist watches. Like the shipping container, file containers can carry many different types of content – the video and audio tracks. These tracks are encoded with some sort of codec. Most codecs compress the video to reduce file size and time to download (and to increase field recording times in production), but there are codecs that work with uncompressed video. Every track has to have a codec for video and for audio.

Common containers are QuickTime (which supports over 160 codecs at last count); AVI (which probably supports almost that many) and MPEG-4, which supports only a few codecs, but very versatile ones. Common codecs are "MPEG-4", "Sorenson", "H.264", "Animation", "Cinepac", etc. (DivX is its own thing, as I'll explain.)

Most QuickTime codecs are for production purposes. The older QuickTime codecs that were used for .mov on the web have been "deprecated" by Apple. They no longer show up as export options in the default install of QuickTime. Nor should they. They're way too inefficient by modern standards. The last new QuickTime distribution codec was Sorenson Video 3 in July 2001. In codec terms that's just a little after the Jurassic era.

AVI has been a workhorse. I refer to it as the zombie format because Microsoft officially killed it in 1996 (when the last development was done). It is still in use in production on PCs and very popular for distribution on the Internet, with more modern codecs in the AVI wrapper. Most AVI production codecs are specific to their hardware parent. A modern .avi file is likely to be a "DivX" file.

DivX is actually a hybrid of an AVI wrapper with an MPEG-4 Advanced Simple Profile (see later) video codec and an MP3 audio track. This is a bad hybrid of codecs and formats, such that DivX for a while had to have their own player. (MPEG-4 video should go with AAC audio in an MP4 container/wrapper.)

Most often the MPEG-4 codecs are used in the MPEG-4 container. This is a modern, standards-based container not owned by any one company. It is an official International Standards Or-

ganization standard. The basic file format was donated by Apple and is heavily based on the QuickTime container, but is NOT the same. You can't just change the .mov to .mp4 (or reverse) and hope it'll work. (It will in the QuickTime player but nowhere else.)

The first codec that the Motion Picture Experts Group (a.k.a. MPEG) approved is properly called MPEG-4 Part 2 'Simple Profile' or 'Advanced Simple Profile.' This was such a great marketing name, that Apple just called it simply "MPEG-4," thereby creating huge confusion for everyone as the distinction between codec and container was totally blurred!

Thanks Apple! Not!

Apple only supported Simple Profile; Sorenson and DivX used Advanced Simple Profile and there were components for QuickTime (not made by Apple) that played Advanced Simple Profile MPEG-4 as well as Simple Profile MPEG-4.

DivX uses the Advanced Simple Profile but in an AVI wrapper, as noted above.

Then just a few years ago, the MPEG association approved a new codec. Called (in full) MPEG-4 Part 10; the Advanced Video Codec (AVC). The European ITU also supported the same codec independent of MPEG-4 (so it could be used in other wrappers) as H.264. They're all the same MPEG-4 codec that is Part 10, Advanced Video Codec or H.264.

And yes it is possible to put an AVC/H.264 video track in a QuickTime .mov, but that's a different container and only QuickTime will play it. MPEG-4 is an ISO standard and there are more than 20 player implementations.

It is AVC/H.264 video with AAC audio (the MPEG audio standard) in an MPEG-4 container that is now playable in QuickTime Player, iTunes, on Apple Devices, in 20 standard players and in Flash 9 release 3 or later (Flash 9r3 was finalized in November 2007 and is now widely installed). Microsoft also support H.264 MPEG-4 in Silverlight 3 (released in 2009), and Windows 9 Media Player has support built in for those same files.

3GPP and 3GPP2 cell phone codecs are part of the MPEG-4 family.