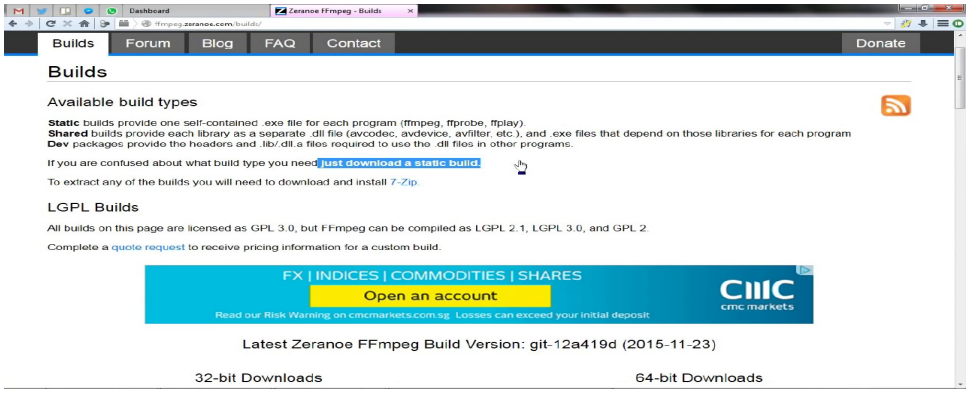
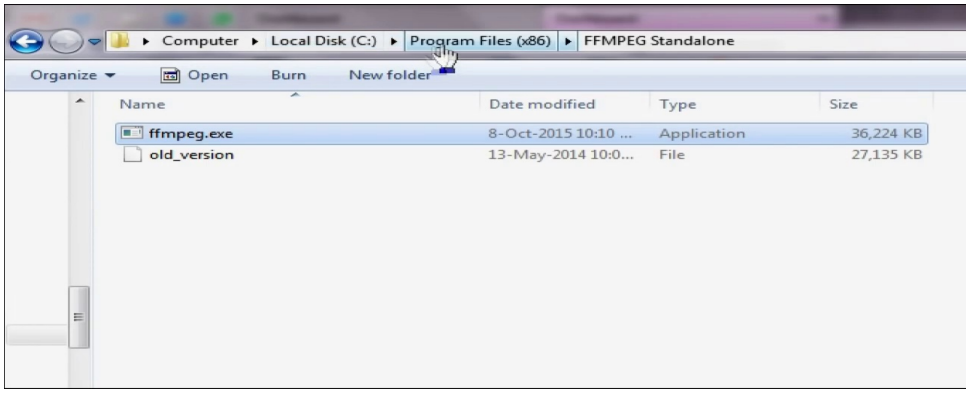
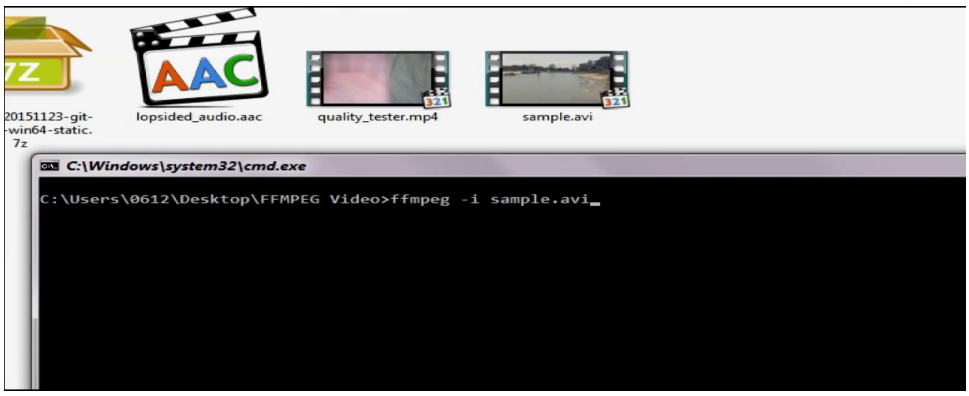


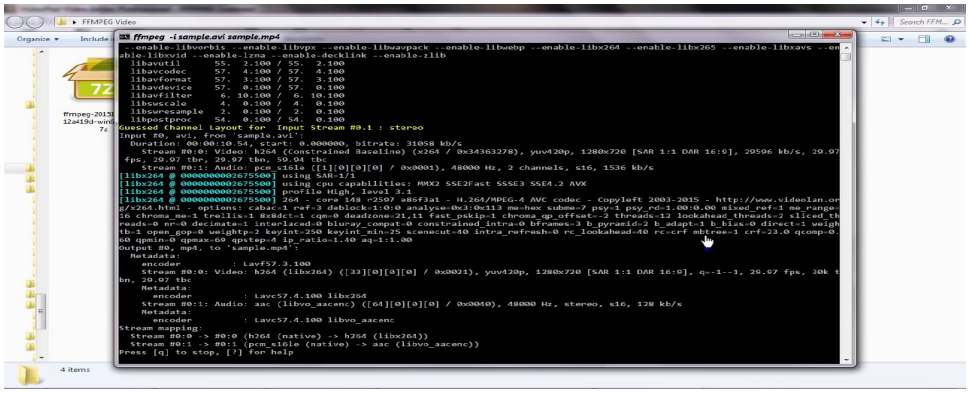
So yeah, I hope this interests you after learning how to use FFM peg. So yeah, I hope you think the same.Hello and welcome back to another random Wednesday episode. Let's begin by learning how to install it. Of course, as it turns out, it is much more than a file format converter.Well, you can find a download link in the video description.You realize right off the bat that things are a little bit complicated because there are three different versions of FF MPEG you can choose from, but they do tell you that you know if you have no idea what you're supposed to do here, just pick the static version so that's what we're going to do. So I would recommend that as well.Is and you realize that what you end up with is an archive. Within the archive, enter the bin folder and look for FF MPEG.In program files folder and then link to it using the path variable.



I do actually have things set up this way because, well, it's just easier for me.With that out of the way, let's move on to the fun part and that is to actually convert files with Ffmpeg. I know I mentioned videos repeatedly, but as it turns out, FM pack can also work with images and audio. You just tell it what follow the process and what kind of output you want and it just does it. So let's actually try a very simple example.Now what you see here is a folder in which I have staged the files I want to work with. Then you say dash I, which means I'm about to specify an input file and then you go ahead and specify the input file.

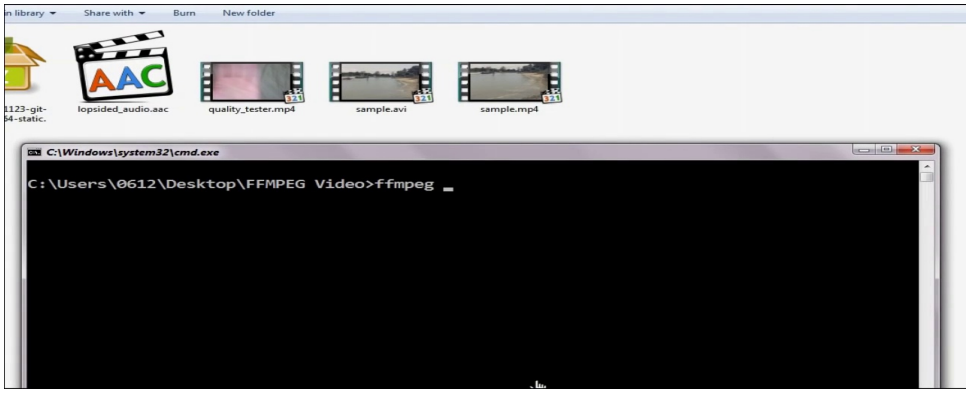


So simply all you have to do is to actually press space and type out the new name of the file, making sure that the extension is MP4, hit enter and well off. FFM Peg is now converting an Avi file to MP4.

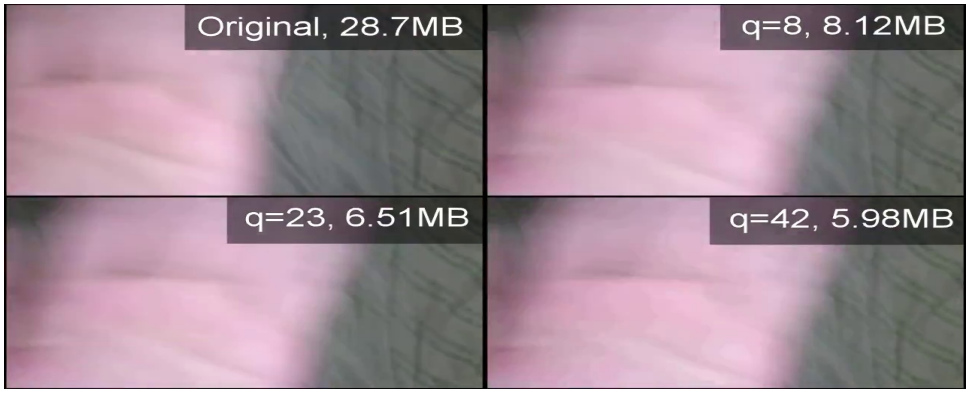


But well, the time will come when you want to do something more complex, and that is where the command switches start to commit. You would probably want to be able to tweak up the quality and what I'm going to show you here works for MP4 and Avi respectively.

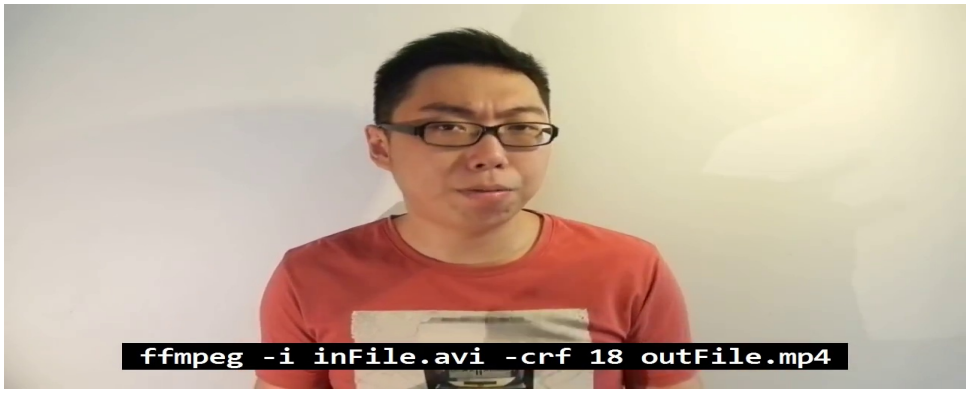
Now there is a factor in image compression called the quantizer that actually affects file quality and it is linked to.Basically, all you need to know is that the quantizer is a number. Of course, the higher the quality, the higher the file size, so there's a tradeoff you need to work with. You specify the input and then you type Q followed by a number.



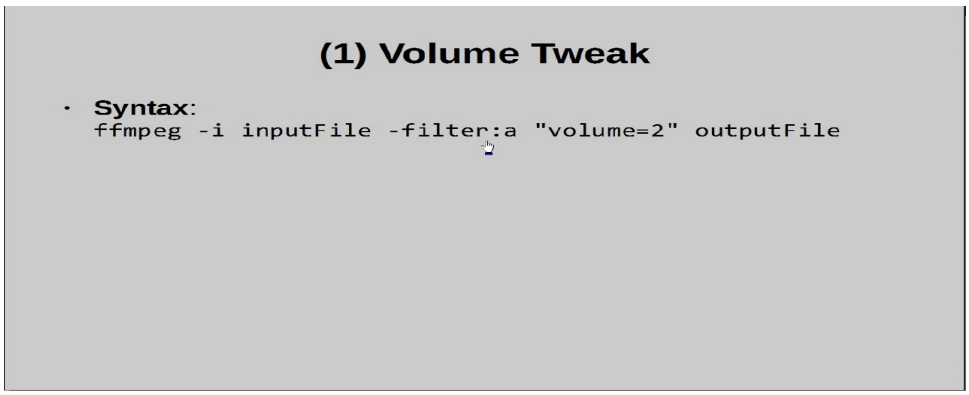
You specify the input and then you type Q followed by a number.



The values are not necessarily the same. Look at the file size.



Now bitrates work for both audio and video in a file, so you'll probably want to tell it whether you want this to work on the video channel or the audio channel.Then give it a number. I can even specify both audio and video bitrates at the same time, and that will fix the two bit rates of the two different streams. So yeah, for the vast majority of use cases, I'll say you know what you've learned so far covers your bases. Let's take a look at Ffmpeg filters just for fun. Here's a deal.5 to go through with you. So yeah, let's take it from the top. Let's quickly jump through the five filters.



The number we provide is actually a multiplier.You can of course use decimal volumes, including values smaller than one to reduce the volume.That's why this command call means the left channel is zero and the right channel is 1.

(2) Channel Remapping

Input Left (0) to output Left (0)

- **Syntax:**
  `ffmpeg -i inputFile -filter:a "channelmap=0-0|0-1" outputFile`

Let's move on to the video filters first, cropping. Notice how we save filter? Because we now want to target video.The first two are with and height, which specified the width and height of the output.

For example, this statement means that we want to crop the video to 2/3 of its original size.

(3) Cropping

- **Syntax:**
  `ffmpeg -i inFile -filter:v "crop=w=2/3*in_w:h=2/3*in_h" outFile`

- Variables in_w and in_h are supplied
  - They stand for input width and input height respectively

Next video scaling. You can also use arithmetic like with cropping the variables in_W and in_H are also available here. Its value will be determined from the length you do provide.

(4) Scaling

- **Syntax:**
  `ffmpeg -i inFile -filter:v "scale=w=640:h=480" outFile`
- **Scaling with arithmetic & variables**
  `ffmpeg -i inFile -filter:v "scale=w=2/3*in_w:h=2/3*in_h" outFile`
- **Proportional Scaling:**
  `ffmpeg -i inFile -filter:v "scale=w=852:h=-1" outFile`

Unfortunately, your angle needs to be specified in radians.

I'll be sure to respond as soon as I can to keep in touch with my future uploads.Check out the official Twitter account for this channel at 0612 TV. Like I said, FF MPEG has a whole host of filters and if you want to look at the full list, I will include the link in the video description so you can find out more. It does look like I've overrun a bit, so sorry about that, but hopefully you found this video insightful. Thank you very much for watching.Consider checking out the rest of my work on my channel. You can find a link to my campaign in the video description.